Escuela Técnica Superior de Ingenieros en Topografía,
Geodesia y Cartografía
Universidad Politécnica de Madrid

Máster en Ingeniería Geodésica y Cartografía

TRABAJO DE FIN DE MÁSTER

ESTUDIO, DEFINICIÓN E IMPLEMENTACIÓN DE UNA ONTOLOGÍA PARA LA CREACIÓN Y CONSULTA DE UNA BASE DE DATOS RELACIONAL SOBRE DATOS OBTENIDOS DEL LINAJE DE LAS PARCELAS CATASTRALES

Alumno: Álvaro Bachiller Hurtado

Tutores: Ramón Pablo Alcarria Garrido
Miguel Ángel Manso Callejo

Madrid, Julio de 2015

Agradecimientos

Gracias a mi familia, sobre todo mi madre y mi hermano que aunque no fueran capaces de entender el proyecto, sí entendían mi necesidad de tiempo y me han ayudado a conseguir todo el que podía descargándome de mis tareas.

Por otro lado quiero agradecer a mis compañeros de máster todos los buenos momentos que hemos compartido y la amistad que hemos forjado. En especial agradecer a Izar, Jacinto y Pablo que compartieran su sabiduría conmigo de la que he aprendido muchísimo y he podido sacar provecho de cada palabra.

También quiero dar las gracias a mis tutores, Miguel Ángel y Ramón, primero por la oportunidad que me han brindado ofreciéndome este proyecto, sacando tiempo de donde no lo hay para poder tutelarme y segundo porque me han apoyado en todo momento y me han guiado por el buen camino, estando siempre disponibles para ayudarme cuando he tenido dificultades.

Finalmente el agradecimiento más especial quiero dárselo a Gabi, por todo el apoyo, cariño y ánimo que me ha dado. Por acompañarme en los momentos difíciles y compartir los buenos momentos. Sin ti esto no habría sido ni la mitad de especial. Gracias.

Hay muchas personas a las que me gustaría agradecer que me hayan acompañado hasta este punto pero no creo ser capaz de acordarme de todos, mis más sinceras disculpas de antemano.

Resumen

En los últimos años la evolución de la información compartida por internet ha cambiado enormemente, llegando a convertirse en lo que llamamos hoy la Web Semántica.

Este término, acuñado en 2004, muestra una manera más "inteligente" de compartir los datos, de tal manera que éstos puedan ser entendibles por una máquina o por cualquier persona en el mundo.

Ahora mismo se encuentra en fase de expansión, prueba de ello es la cantidad de grupos de investigación que están actualmente dedicando sus esfuerzos al desarrollo e implementación de la misma y la amplitud de temáticas que tienen sus trabajos.

Con la aparición de la Web Semántica, la tendencia de las bases de datos de nueva creación se está empezando a inclinar hacia la creación de ontologías más o menos sencillas que describan las bases de datos y así beneficiarse de las posibilidades de interoperabilidad que aporta.

Con el presente trabajo se pretende el estudio de los beneficios que aporta la implementación de una ontología en una base de datos relacional ya creada, los trabajos necesarios para ello y las herramientas necesarias para hacerlo.

Para ello se han tomado unos datos de gran interés y, como continuación a su trabajo, se ha implementado la ontología. Estos datos provienen del estudio de un método para la obtención automatizada del linaje de las parcelas registradas en el catastro español.

Abstract

In the last years the evolution of the information shared on the Internet has dramatically changed, emerging what is called Semantic Web.

This term appeared in 2004, defining a "smarter" way of sharing data. Data that could be understood by machines or by any human around the world.

Nowadays, the Semantic Web is in expansion phase, as it can be probed by the amount of research groups working on this approach and the wide thematic range of their work.

With the appearance of the Semantic Web, current database technologies are supported by the creation of ontologies which describe them and therefore get a new set of interoperability possibilities from them.

This work focuses in the study of the benefits given by the implementation of an ontology in a created relational database, the steps to follow and the tools necessary to get it done.

The study has been done by using data of considerable interest, coming from a study of the lineage of parcels registered in the Spanish cadaster. As a continuation of this work an ontology has been implemented.

Nota para leer el presente trabajo de fin de máster:

Los símbolos que se encuentran en el encabezado y el pie de página son enlaces que llevan al lector a diferentes partes importantes del documento:

Para acceder al índice desde cualquier punto del documento pulse 

Para regresar al principio de una sección 

Acceso al índice de figuras 

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1. Introducción

Los bienes inmuebles según el Diccionario de la Real Academia de la lengua Española (RAE, 2015) son:

Tierras, edificios, caminos, construcciones y minas, junto con los adornos o artefactos incorporados

En el territorio español, están descritos en un registro administrativo denominado Catastro. Este archivo alberga una gran base de datos que, por la naturaleza de los mismos, tiene un carácter dinámico, albergando los continuos cambios que se dan en las unidades elementales del registro: las parcelas catastrales.

Catastro era una base de datos estática en la que se registraba una instantánea del estado del territorio Español. Las diferentes modificaciones y variaciones de dichas parcelas también modificaban el registro de la base de datos, sobrescribiendo los registros anteriores (Moreno et al., 2014). A partir de 2001, las diferentes alteraciones de las parcelas quedan guardadas como una nueva versión de la misma (quedando la estructura como una sucesión de instantáneas).

La información histórica y el linaje de las parcelas, por razones evidentes, son de gran interés para muchas organizaciones (administraciones, autoridades legales, propietarios, etc.), por lo que, para almacenar la historia de las entidades y poder hacer el seguimiento de la evolución de las mismas, es necesario también registrar las relaciones existentes entre ellas a lo largo del tiempo (Moreno et al., 2014).

El trabajo de Moreno et al. presenta un método para obtener las relaciones de linaje más frecuentes entre las parcelas (agregación y segregación) y propone un prototipo de estructura relacional para el almacenamiento y la gestión histórica de los datos catastrales de acceso público.

Otro de los inconvenientes de Catastro es que no existe una definición del ser aceptada ni estandarizada. Una ontología podría aportar una interoperabilidad mejorada e integrar diferentes fuentes de datos (Agarwal, 2005). Del mismo modo, podría aumentar los beneficios que aporta el prototipo de Moreno et al.

Las ontologías describen sistemáticamente una realidad mediante una serie de axiomas lógicos (Guarino, 1998). Son clave en la web semántica, donde facilitan la interoperabilidad, especificando un vocabulario común en un lenguaje de ontología (Fu et al., 2005).

Habitualmente, los estudios se han centrado en la implementación de ontologías "superiores", pero no en la profundización de los conceptos generales subyacentes (Gantner, 2011).

En concreto, cuando se hace el estudio de ontologías superiores sobre divisiones administrativas geográficas, a pesar de ser un campo muy extendido, se comprueba que apenas existen (Lacasta et al., 2014).

Estudios como el presente Trabajo de Fin de Master, buscan, crean y/o adaptan una ontología para adecuarla a una superior. Esta ontología debe unir los conceptos subyacentes de la fuente de datos con la ontología superior y capacitar al usuario para hacer consultas espaciotemporales.

Se pretende implementar una ontología y un nuevo modelo de datos que ajuste el sistema general de base de datos relacional sobre el linaje catastral propuesto por Moreno et al., que reporte un aumento de los beneficios que ya aporta a Catastro.

2. Objetivos

El objetivo principal de este trabajo es demostrar los beneficios de las tecnologías *Open Data* y *Linked Data* frente al trabajo necesario para la transformación de una base de datos con información de parcelas catastrales a una base de datos abiertos y enlazados que puede ser consultado mediante consultas semánticas SPARQL.

Con el presente estudio se pretende profundizar en el Open Data, así como en las ontologías y el *Protocol And Rdf Query Language* (SPARQL) aplicados a la explotación en forma de consulta de las relaciones de linaje entre parcelas catastrales. Se desarrollarán de esta manera los conocimientos obtenidos en el Máster de Ingeniería Geodésica y Cartografía dentro del marco de las asignaturas: Bases de Datos Espaciales; Infraestructura de datos espaciales y Programación Web.

Mediante este estudio e implementación se pretende demostrar los beneficios que pueden aportar las ontologías, cuya aplicación permite dar un valor añadido a la información de linaje que, almacenada en un *Sistema de Gestión de Base de Datos Relacional* (SGBDR, RDBMS en sus siglas en inglés), se convierte en tripletas sujeto-objeto-predicado sobre las que aplicar consultas en un lenguaje estandarizado: SPARQL.

La base de datos relacional de linaje catastral requerirá de un estudio y un conocimiento profundo de los datos antes de la búsqueda de la ontología, ya que sería paradójico crear una ontología sin conocer lo que se quiere definir. Primeramente se utilizará Geokettle para georreferenciar la base de datos cargarla a PGAdminIII donde se realizará la manipulación y estudio de la base de datos mediante consultas Structured Query Language (SQL).

Para la creación de la ontología se pretende buscar una ontología superior que se adecúe a los datos. La adaptación de la misma, si fuera necesario, se realizará con el programa PROTÉGÉ.

Una vez definida la ontología, se procederá a su carga en PROTÉGÉ y a la creación en este programa de una serie de ejemplos para comprobar su funcionalidad, validez y adecuación a los datos de linaje.

Finalizadas las pruebas realizadas a los ejemplos, se procederá a la creación de las series de tripletas que definirán la base de datos relacional en el lenguaje *Resource Description Framework* (RDF). No existe ningún programa que, de manera automática, sea capaz de transformar una base de datos en series de tripletas, por lo que será necesaria la creación de un programa Java diseñado explícitamente para el estudio. Dicho programa será capaz de leer la base de datos en formato *Comma Separated*

Values (CSV) y escribirá las tripletas, con el diseño ontológico seleccionado, en formato RDF.

El SGBDR creado se cargará en un servidor local creado por la aplicación Virtuoso, para así poder hacer una serie de consultas SPARQL que se contrastarán con consultas equivalentes en SQL con la base de datos previa. Con estas pruebas se pretende demostrar la superioridad operacional y procesal de una base de datos relacional frente a su predecesora.

Finalmente se creará una aplicación web que muestre los resultados con algunos ejemplos de consultas útiles a la base de datos para mostrar, en toda su extensión, la gran rentabilidad que otorga el trabajo de aplicar una ontología a una base de datos relacional.

3. Estado del arte

A continuación se pasa a describir los antecedentes que se deben de tener en cuenta para la realización de este estudio.

3.1. Catastro

El Catastro, es un registro administrativo dependiente del Ministerio de Economía y Hacienda donde se describen los bienes inmuebles (tanto urbanos, como rústicos y de caracteres especiales). Está regulado por el "Texto Refundido de la Ley del Catastro Inmobiliario" donde se estipula que la inscripción de los bienes inmuebles (y las modificaciones de sus características) en la base de Catastro es obligatoria y gratuita, características que lo diferencia del Registro de la Propiedad (Catastro, 2007).

Las características básicas recogidas en la base de catastro son:

- Ubicación, coordenadas bidimensionales georreferenciadas que describen la posición del inmueble
- Dimensiones, en metros cuadrados
- Uso, urbano, rústico o caracteres especiales
- Propietarios, persona física, persona jurídica, uso común, estatal…
- Derechos, servidumbres, hipotecas...
- Referencia catastral, identificador único y obligatorio asociado a cada bien inmueble
- Dos marcas temporales, definen el período de validez

Gracias a los valores de la referencia catastral y de ubicación se pueden distinguir las parcelas de manera unívoca. Si junto a estos datos se consultan las marcas temporales se podría hacer un seguimiento de su linaje o historia.

Tras la resolución del 23 de marzo de 2011 los datos de Catastro son accesibles al público a través de su Oficina Virtual. Sin embargo, no se puede acceder a la información sobre la genealogía o las relaciones de linaje entre las parcelas, de forma que la gestión de la información histórica es muy limitada. Si un usuario necesita datos de la evolución de un conjunto específico o bien de todas las parcelas de un municipio tendría que diseñar un proceso de técnicas de análisis que no está descrito ni validado por catastro en ningún trabajo de investigación (Moreno et al., 2014).

Como funciones principales de catastro se pueden citar:

- Mantener la seguridad jurídica del derecho de propiedad a través de la aprobación y archivo de las mensuras (la base de las escrituras de traslación y dominio)
- Cálculo de contribuciones como el impuesto inmobiliario
- Descarga masiva de información catastral (disponible desde abril de 2011); siendo muy utilizado por Administraciones, ciudadanos y empresas
- Base para el planeamiento urbano y rural

La incorporación de los bienes a Catastro y las alteraciones de sus características que, como ya se ha enunciado anteriormente, es obligatoria, suele ocurrir de las siguientes formas:

- Declaraciones, comunicaciones y solicitudes
- Subsanación de discrepancias
- Inspección catastral: actuaciones de comprobación e investigación de hechos, actos, negocios… Que pueden originar una incorporación o modificación en el Catastro (de información, de valoración y de informe, asesoramiento…)

La creación y mantenimiento del Catastro, así como su la difusión de su información, es competencia exclusiva del Estado. La inspección, elaboración y gestión de la cartografía catastral, las ejercerá la Dirección General del Catastro (DGC) o se realizará a través de colaboraciones de otras Administraciones, entidades y corporaciones locales. La valoración queda como competencia exclusiva de la DGC.

3.2. *Open Data y Open Linked data*

Linked data hace referencia a las prácticas recomendadas para crear, compartir e integrar conjuntos de datos en la Web Semántica.

Se base en cuatro principios:

- El uso de URI (*Uniform Resource Identifier*) para identificar elementos o conceptos de forma unívoca, por ejemplo:

http://dbpedia.org/describe/?url=http%3A%2F%2Fdbpedia.org%2Fresource%2FCatastro&sid=144629

- URI con el protocolo de acceso HTTP (*HyperText Transfer Protocol*)
- Ofrecer información sobre los recursos con el lenguaje RDF (especificación de la W3C (World Wide Web Consortium) creado para escribir metadatos.

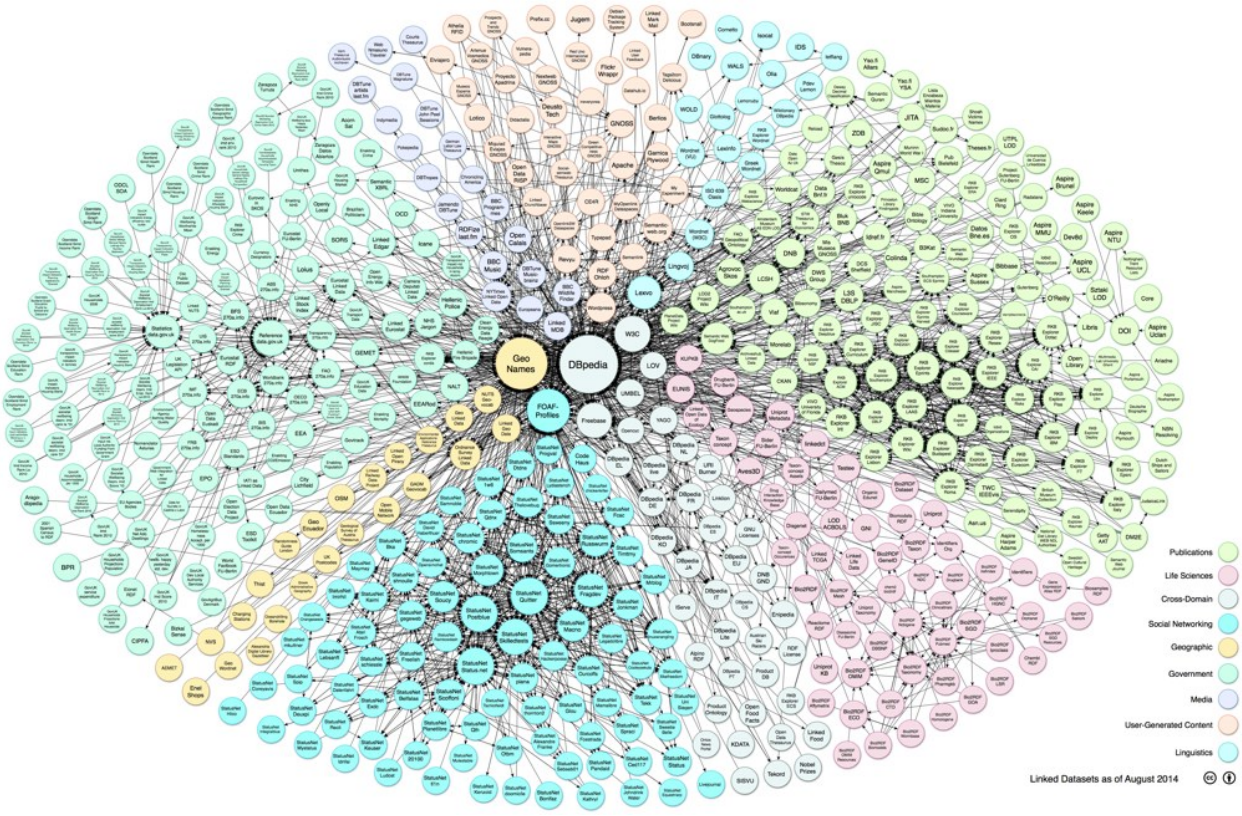

Figura 1. ***Linked Open Data Cloud Diagram*****, esquema de** ***Linked data***

En la actualidad se usa como método general para la descripción de conceptos o recursos web)

- Incluir enlaces a otros elementos (URI)

Los datos semánticos se utilizan mediante lenguajes de consulta de datos enlazados. El más extendido es el SPARQL, que está estandarizado para realizar consultas a documentos RDF y normalizado por el DAWG (*RDF Data Access Working Group*) del W3C.

La aplicación en el mundo geográfico del *linked data* (Figura 2) une la información espacial como otra propiedad RDF, lo cual requiere una definición de un esquema RDF y una extensión de los lenguajes de consulta para ellos.

Para conseguir este objetivo existen varias iniciativas, tales como stSPARQL o W3C GEO *vocabulary*. El más extendido hasta la fecha es GEOSPARQL: un estándar para la representación y consulta de *linked data* propuesto por el *Open Geospatial Consortium* (OGC). Enumera las propiedades y relaciones espaciales necesarias para tratar datos en formato GML (*Geography Markup Language*) o WKT (*Well Known Text*), extendiendo, además el lenguaje SPARQL para consulta de datos geográficos. Se basa en las normas:

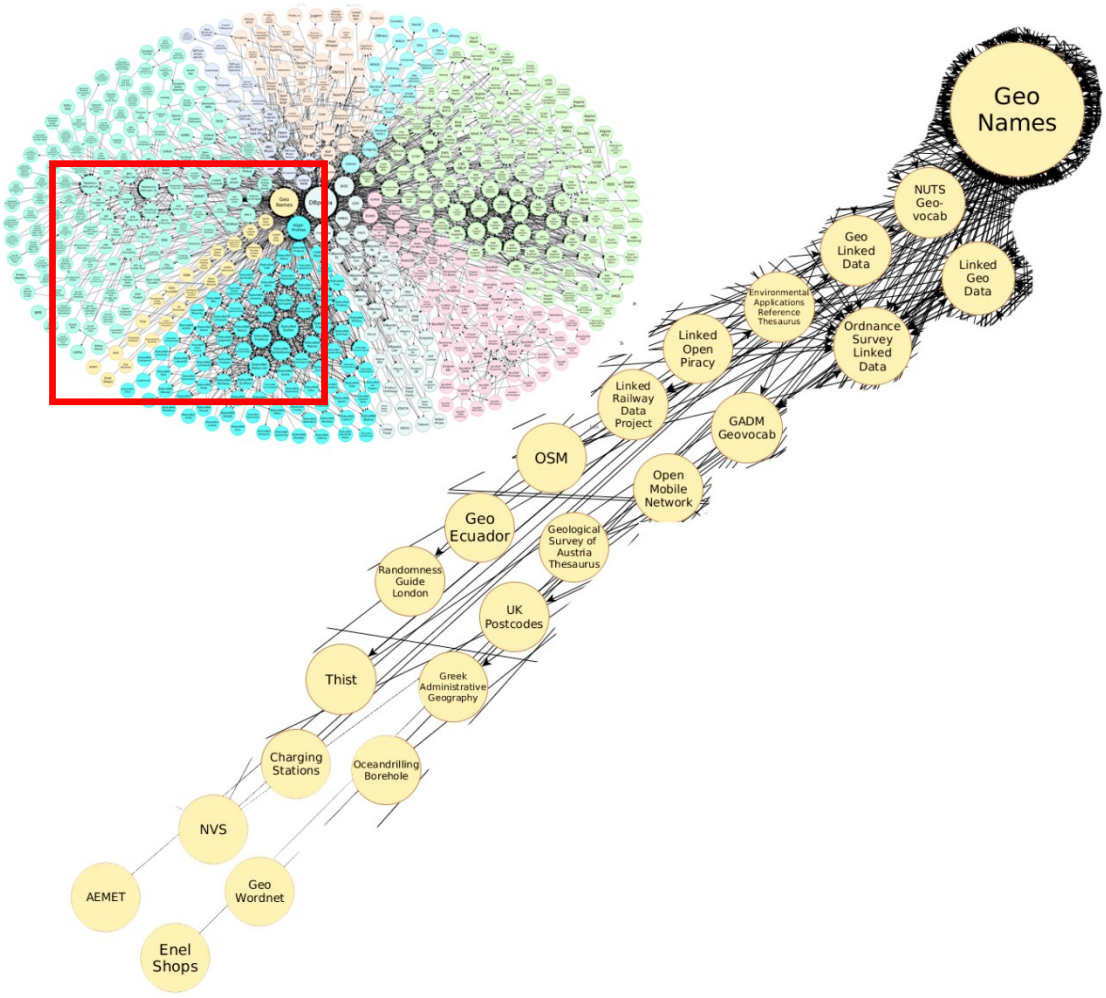

Figura 2. Detalle de la parte geográfica en Linked data

- ISO 19109: *Rules for application schema*
- ISO 19125: *Simple Features*
- ISO 19107: *Features Geometry*
- ISO 12349: *Spatial SQL database multimedia languages*

Open data es una filosofía que pretende que determinada información esté disponible de manera "abierta" (accesible) para todo el mundo, sin tener que lidiar con derechos de autor, patentes u otros mecanismos de control. Esta ética es común encontrarla también en el *software* libre, el código abierto o los estándares abiertos.

Un dato libre es aquel que puede ser utilizado, reutilizado y redistribuido sin trabas por cualquier persona. Es común encontrar estos datos sujetos al requerimiento de atribución de credenciales y de compartición en la misma manera (licencias *Creative Commons*: CC-by-sa).

Open Linked Data, surge de la unión de estos dos conceptos: datos abiertos en formato RDF. Un usuario sería capaz de enlazar datos provenientes de diversas

fuentes, instituciones u organizaciones, explorar y combinar estos datos de manera libre, sin problemas de *copyright*.

Algunos ejemplos de iniciativas *Open Linked Data* son:

- *SmartOpenData*
- *DBpedia*
- *Geonames ontology*
- *GeoLinkedData*
- AemetLinkedData

Se cree que esta filosofía encaja perfectamente con datos de origen público, por ello se pretende implementar sobre unos datos de origen catastral.

3.2.1. Web Semántica

A lo largo de la historia la web ha ido evolucionando. Primeramente existía la web sintáctica (también llamada web 1.0) compuesta por una serie de recursos estáticos, enlazados entre sí. Posteriormente la web se empezó a construir gracias al trabajo de todos los usuarios (foros, blogs, redes sociales…), convirtiéndose en la web colaborativa o web 2.0.

Estos tipos de webs están compuestos principalmente de documentos HTML en lenguaje natural y multimedia. Sus principales característica son: no enlaza todas las páginas existentes; tiene una alta sensibilidad a las palabras utilizadas en la búsqueda y escasa precisión cuando se busca un dato concreto.

La web semántica, o web 3.0, surge con la nueva idea de que se puede encontrar

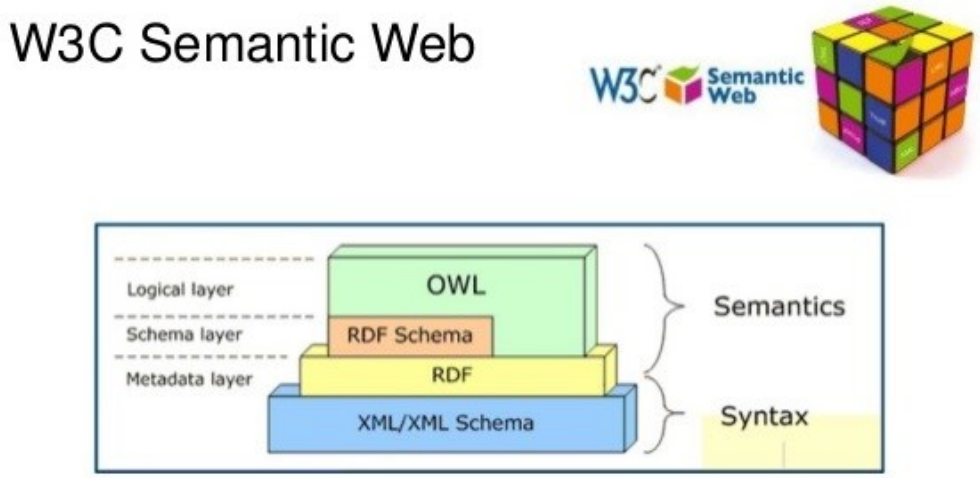

Figura 3. Esquema básico de la web semántica

informaciones determinadas o integrarlas. En esta web se añade la semántica que le falta a la web sintáctica, creando un entorno donde se puede acceder a la información deseada de manera exacta y completa. Se facilita así el procesado de información y la resolución de problemas de interoperabilidad entre aplicaciones (W3C, 2004).

La búsqueda de información se realiza gracias a la utilización de una infraestructura común, mediante la cual, es posible compartir, procesar y transferir información de forma sencilla. Se apoya en lenguajes universales que resuelven problemas ocasionados por una Web carente de semántica (W3C, 2004). Este lenguaje universal permite el intercambio de datos y brinda un mayor significado a los mismos, pudiendo así ser interpretados por las máquinas.

3.2.2. Ontología

La definición del concepto Ontología en este dominio de aplicación actualmente más utilizada es la de Gruber (2009):

"En el contexto de los ordenadores y las ciencias de la información, una ontología define un grupo de primitivas representativas con las que modelar un dominio o discurso de conocimiento. […] una ontología puede ser vista como un nivel de abstracción de los modelos de datos, análogamente a modelos jerárquicos y relacionales, pero destinados a modelar el conocimiento sobre individuos, sus atributos, y sus relaciones con otros individuos."

Estas primitivas suelen ser clases (grupos), atributos (propiedades) y relaciones, incluyendo información sobre su significado y sus restricciones que la hacen consistente.

Los lenguajes de las ontologías son más cercanos en expresividad a la lógica que a los lenguajes usados para modelar bases de datos tradicionalmente. Por esta razón se dice que las ontologías están descritas a nivel "semántico", mientras que el esquema de base de datos es un modelo de datos a nivel "lógico" o "físico". Gracias a esto datos de origen heterogéneo pueden ser exportados, trasladados, consultados y unificados a través de sistemas y servicios desarrollados independientemente. Las aplicaciones que han conseguido esto hoy en día son capaces de ofrecer interoperabilidad entre bases de datos, búsqueda cruzada en las bases de datos e integración de servicios web (Gruber, 2009).

Por otro lado, la Web Semántica también usa "vocabularios", considerados como una forma especial (normalmente una versión ligera) de ontología que, a veces, simplemente es una colección de URI con un significado descrito. Los vocabularios son los cimientos para las técnicas de inferencia en la Web Semántica (W3C, 2008).

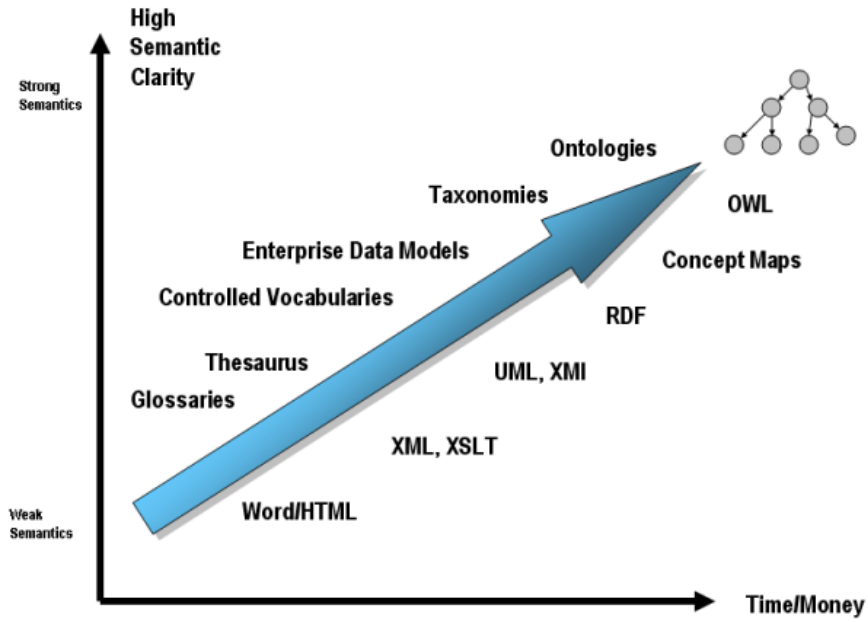

Figura 4. Evolución de la descripción del conocimiento

La W3C en su definición de ontología (*Ontology*, 2008), advirtió que la línea entre "ontología" y "vocabulario" es difusa. Los vocabularios pueden llegar a ser muy complejos en su descripción de conceptos y relaciones, variando desde uno o dos conceptos a varios miles de términos. La tendencia seguida hasta ahora ha sido usar *ontología* para colecciones de términos más complejas y formales; *vocabulario* queda relegado a definiciones en las que no es necesario el uso de un formalismo tan estricto.

En resumen, las ontologías son esquemas conceptuales, rigurosos y exhaustivos, que forman parte de los estándares W3C para la Web Semántica, donde son usados para hacer una especificación de vocabularios estándar con los que relacionar datos entre sistemas, dar servicio para responder consultas, publicar bases de datos reusables y ofrecer servicios para facilitar la interoperabilidad entre múltiples y heterogéneos sistemas y bases de datos.

Debido a los siguientes beneficios se ha decidido crear una base de datos relacional estudiando, definiendo e implementando una ontología para los datos obtenidos del linaje de las parcelas catastrales:

- La inclusión de carácter semántico a una base de datos permite una mayor organización de la información y una mayor definición (se realiza a través de conceptos)
- Las búsquedas de contenido relevante se realizan por significado y no por contenido textual

- Sencillez de procesos gracias a la interoperabilidad e integración de bases de datos

3.2.2.1. Buscadores de ontologías

También llamados "motores de búsqueda de ontologías", funcionan de manera similar a los buscadores de contenido que se usan a diario en todo el mundo (Google, Bing, Yahoo…). Realizan una búsqueda mediante un algoritmo (que difiere según el buscador) a una base de datos propiamente indexada. Gracias a este algoritmo y a los datos históricos de búsqueda, los resultados se pueden mostrar al usuario por orden de relevancia.

Por otro lado al ser un motor de búsqueda tradicional, se pierden las ventajas que aporta la Web Semántica, que es, finalmente, el objeto de este estudio.

Debido a estas razones no se han visto de utilidad los buscadores consultados y se han descartado para su uso en el desarrollo del proyecto.

3.2.2.1.1. *Linked Open Vocabularies* (LOV)

Reconocido por instituciones científicas de primer orden como: W3C, Datalift, Joinup (creado por la comisión europea); o compañías multinacionales como: Mondeca (compañía de software especializada en tecnologías semánticas) y Fujitsu (LOV, 2013). Es uno de los centros de información más importantes en el campo de los vocabularios.

Con 512 vocabularios indexados, su algoritmo no solo tiene en cuenta la popularidad temporal, sino también la las puntuaciones asignadas por LOV (2013) y el grado de coincidencia con el término buscado.

Realiza las búsquedas en cuatro niveles diferentes:

- *local name*
- *primary labels*
- *secondary labels*
- *tertiary labels*

Basándose en el nivel donde encuentre la coincidencia con el término el motor de búsqueda asignará una puntuación de relevancia a la ontología encontrada.

Debido a la calidad y precisión de los resultados obtenidos este buscador ha sido el más utilizado de los que se describen en el presente estudio.

Figura 5. Página principal de Swoogle

3.2.2.1.2. *Swoogle*

Buscador en vías de desarrollo, fue creado en 2005 por un grupo de investigación del departamento de informática e ingeniería eléctrica de la universidad de Maryland (Swoogle, 2005).

Su página principal tiene una sencillez que recuerda a Google (Figura 5). Este buscador se encarga de utilizar su algoritmo en internet buscando un tipo especial de documentos: *Semantic Web documents*, escritos en RDF para ofrecer los siguientes servicios para la Web Semántica:

- Búsqueda de ontologías
- Búsqueda de tipos de datos
- Búsqueda de sujetos (URI, por ejemplo)
- Búsqueda de metadatos
- Guardado de diferentes versiones de documentos

En la práctica los resultados de este buscador eran demasiado genéricos y poco específicos, por lo que se descartó como herramienta de trabajo para este estudio.

3.2.2.1.3. *Watson*

Interfaz web de gran sencillez para un motor de búsqueda que ofrece unos resultados bien estructurados y precisos.

Sus principios de diseño se basan en:

- Centrado en la calidad semántica
- Explorar relaciones entre ontologías

- Proveer un acceso a los datos rico y semántico

Fue desarrollada parcialmente gracias al apoyo de los proyectos NeOn y OpenKnowledge, que usaban fondos de la comisión europea.

Un resumen de sus funcionalidades puede verse en la Figura 6.

Su algoritmo realiza búsquedas en documentos semánticos donde las palabras clave introducidas aparezcan como identificadores, clases, propiedades o individuos. Las opciones de búsqueda permiten restringir la búsqueda a unas entidades en concreto o, incluso, a elementos precisos dentro de las propias entidades (nombre local, etiqueta, comentario o cualquier literal) (Watson, 2008). Como resultado se obtiene una lista de URI's (o series de URI si hay información enlazada en los mismos) con una pequeña cita del documento encontrado (igual que los buscadores tradicionales).

En comparación con el resto de los buscadores es uno de los más funcionales, que aportaban una información de gran utilidad con URI's en las que profundizar. Debido a esto es uno de los más utilizados en el presente estudio.

3.2.2.1.4. SPIRIT

SPIRIT (*Spatially-Aware Information Retrieval on the Internet*) fue un proyecto de investigación que busca diseñar e implementar un motor de búsqueda que encuentre documentos y bases de datos en relación a lugares o regiones referidas en una consulta (SPIRIT, 2007). El proyecto ha sido desarrollado en colaboración de las universidades de: Cardiff, Hannover, Sheffield, Utrecht, Zürich y el Institut Géographique National francés.

Ha creado varias herramientas web y técnicas que pueden ser usadas para crear

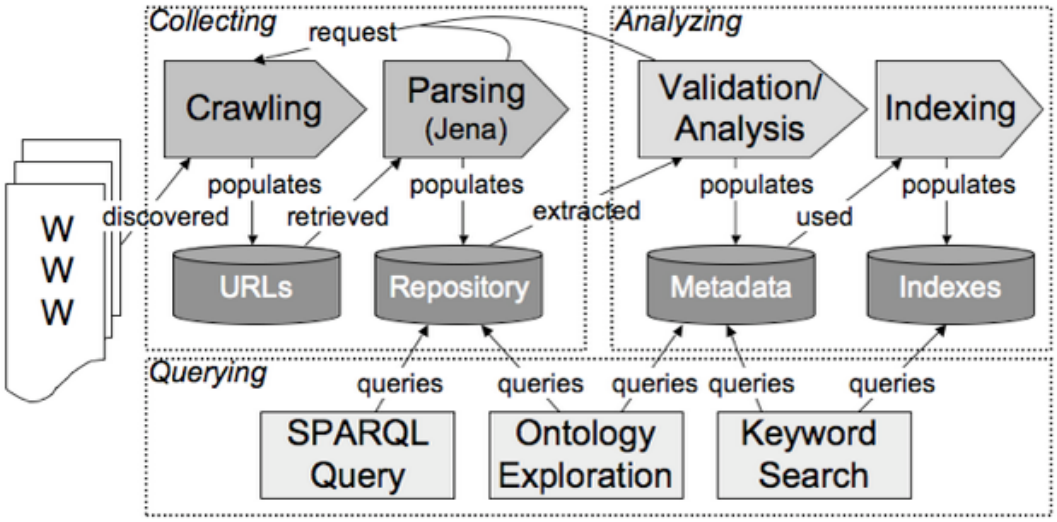

Figura 6. Funcionalidades de Watson

buscadores y páginas web con capacidad para el reconocimiento de terminología geográfica.

El objetivo final de este proyecto es el de crear un motor de búsqueda que sea capaz de reconocer la localización, encontrando documentos y bases de datos relacionados exactamente (o aproximadamente) a los lugares o regiones referidos en la consulta.

A pesar del gran interés que puede suscitar este proyecto, se encuentra parado en la actualidad y el motor de búsqueda sólo está en fase de desarrollo.

3.2.2.2. Listados de bibliotecas

Existe un gran número de páginas, llamadas "listado de ontologías" o "bibliotecas de ontologías" (el término "*library*" se usa habitualmente de manera ambigua para los conceptos de "listado" y de "ontología"), que hacen referencia a ontologías simplemente uniéndolas por su temática a una URI.

Según la W3C (2013), este tipo de páginas requieren un mínimo de esfuerzo: si los datos de los autores están dentro de los dominios referenciados en la lista, la ontología correspondiente puede ser usada.

Estas listas conllevan una serie de cuestiones: "¿Cómo definir qué está en la lista?". La popularidad es uno de los aspectos, la calidad puede ser otra, pero ¿Qué es una ontología de calidad? ¿Cuándo se convierte en popular? ¿Quién lo decide?

Más adelante, este estudio tratará de contestar, al menos, alguna de estas cuestiones.

Los listados de ontologías han demostrado ser de gran utilidad para el estudio e implementación de las mismas en los estudios, prueba de ello es el gran número de listados que se pueden encontrar a disposición de los usuarios de manera libre.

Al estar agrupadas las ontologías de manera temática, la búsqueda de un conjunto de objetos, definiciones y propiedades, se realiza de manera más rápida y eficaz que si se hace término a término (como ocurre con los buscadores de ontologías). Debido a esto se ha dedicado más tiempo al estudio de los listados en el presente trabajo que a los motores de búsqueda.

Para que esta búsqueda sea más útil y efectiva, es necesario un conocimiento profundo de los datos a los que se pretende aplicar la ontología.

Durante el estudio previo fueron de mucha más utilidad los listados de ontologías que los buscadores, debido a ello se hizo mayor hincapié en éstos. Finalmente, debido a los requerimientos del estudio, tampoco se utilizaron los listados de ontología en la fase de creación de la ontología.

3.2.2.2.1. *Semantic Web*

Semantic Web se define a sí mismo en su página principal como una extensión del *World Wide Web* que permite a las personas compartir contenido más allá de las limitaciones de las aplicaciones y los sitios web.

Implica a una gran cantidad de investigadores, que se encargan de crear tecnologías semánticas de vanguardia, agrupándoles en el su portal en forma de comunidad.

Dentro de este proyecto de investigación se encuentran ontologías que son básicas hoy en día tanto para la aplicación de las ontologías como para hacer consultas a bases de datos relacionales (*Dublin Core*, FOAF…), por tanto este listado es de utilizad para sentar la base de la ontología y las consultas a desarrollar.

3.2.2.2.2. *COLORE*

La universidad de Toronto (Canadá) creó el laboratorio de tecnologías semánticas, de cuyo trabajo surgió COLORE, un listado de los más extensos y completos que se pueden encontrar en la red.

En su página (COLORE, 2009) enuncia que uno de los principales obstáculos para obtener una ontología de varios dominios es la falta de ontologías genéricas adecuadas para especificar la semántica de los conceptos primitivos.

El objetivo de COLORE es construir un repositorio abierto de ontologías superiores que servirán como campo de pruebas para las técnicas evaluación e integración de ontologías. Como valor adicional del proyecto, se cita que se podrán desarrollar nuevas ontologías.

Debido a la gran variedad de ontologías que presenta y la amplitud de campos que contempla, COLORE debe ser una de las primeras referencias a consultar cuando se crea una base de datos relacional.

3.2.2.2.3. DAML

DAML (*DARPA Agent Markup Language*) es un programa que empezó en agosto del año 2000 en una reunión en Boston e integraba a 22 equipos de investigación. Pretende crear un lenguaje y unas herramientas que faciliten el concepto de Web Semántica.

Su listado cuenta con 282 ontologías, de las cuales se pueden encontrar muchas con componente geográfica que son interesantes para el estudio por su definición de las características de las superficies y de las divisiones administrativas.

3.2.2.2.4. NeOn

NeOn es un proyecto en el que participan 14 instituciones europeas, cofinanciado por la comisión europea. Se inició en marzo de 2006 y su objetivo es avanzar en el uso de ontologías en aplicaciones semánticas de gran escala en las organizaciones implicadas. Concretamente pretende mejorar la capacidad de manejar ontologías de varias fuentes, que existen en un contexto particular, que son creadas de manera colectiva y pueden ser muy dinámicas y en constante evolución (NeOn, 2010).

Gracias a las tecnologías y métodos desarrollados por NeOn se crearon una serie de redes de ontologías que pretenden ser de utilidad como esqueleto para las aplicaciones construidas en los casos de estudio del proyecto. Estas redes incluyen ontologías creadas por requerimiento específico de una aplicación y ontologías existentes, rehusadas e integradas con las redes (NeOn Project, 2010).

Cuenta con, aproximadamente, 26 ontologías de origen e índole muy variada de gran utilidad para un amplio abanico de campos. En el campo de la geografía se encuentran: *Geopolitical Entities* y *FAO Geopolitica Ontology*.

3.2.2.2.5. Otros listados

Como ejemplo del alcance de las ontologías se citarán otras de las encontradas, pero que tuvieron menos influencia en el estudio:

- *MapOnto* (http://www.cs.toronto.edu/semanticweb/maponto/ontologies.html), el proyecto principal de este estudio de la universidad de Toronto (Canadá) es la realización de una herramienta que sea capaz de realizar búsquedas de mapas semánticos para mostrar al usuario resultados lo más precisos posible. Como apoyo a este proyecto se realizó una pequeña biblioteca de ontologías que no fue de utilidad para este estudio
- *Ontaria* (http://www.w3.org/2004/ontaria/), que desde otoño de 2004 se encuentra parada, se centraba en vocabularios con ontologías en OWL
- Existen varios listados de ontologías escritas en Protégé, creadas por los usuarios, a pesar de estar disponibles cuando se realizó esta documentación, en la actualidad no se pueden encontrar en los enlaces referenciados

3.2.2.3. Ontologías superiores

La W3C en 2008 define ontología como una definición de conceptos complejos, abstractos y formales.

Gantner, en su estudio de 2011 (Gantner 2011), hace referencia al término "ontologías superiores", que no se ha encontrado en ninguna otra referencia, como el eje central de la investigación ontológica. Tienen como contrapartida que no profundizan en los conceptos generales subyacentes.

La coincidencia entre ambas definiciones es bastante evidente, más aún si se tiene en cuenta la definición de vocabulario dada por la W3C y se compara con los conceptos subyacentes a los que hace referencia Gantner.

A continuación se describen una serie de ontologías básicas que definen los pilares de cualquier ontología que se quiera definir en la actualidad.

Del mismo modo que ocurrió con los listados de ontologías, una vez finalizada la fase de estudio de las ontologías superiores se descartó su uso para la creación de la ontología del presente proyecto ya que, a pesar de su utilidad al aportar las líneas generales sobre las que trabajar, su aplicación es demasiado general para el grado de concreción que se desea alcanzar en el presente estudio.

3.2.2.3.1. RDF *Schema*

La W3C, publicó su versión 1.1 el 25 de febrero de 2014.

Pretende dar soporte a las ontologías definiendo un vocabulario de modelado de datos en RDF. Es una extensión del vocabulario básico de RDF.

3.2.2.3.2. XML *Schema*

También creado por la W3C, su última recomendación está fechada en el 5 de abril de 2012. Facilita la descripción de la estructura y limita los contenidos de los documentos XML.

3.2.2.3.3. LOCN

Su segunda versión ha sido publicada el 23 de marzo de 2015 y pertenece a W3C.

Enuncia un listado con un mínimo de clases y propiedades para describir cualquier lugar (nombre, dirección o geometría). Está diseñado específicamente como ayuda para la publicación de datos y hacerlos así interoperables dentro de la directiva europea INSPIRE.

3.2.2.3.4. Persistent Uniform Resource Locators (PURL)

Actúa como un identificador permanente de direcciones web en un mundo dinámico y constantemente cambiante. En lugar de referenciar directamente las páginas crea un puente que permite a las páginas subyacentes cambiar a lo largo del tiempo sin afectar negativamente a su ontología.

Está estructurado como una comunidad de investigadores que pueden manipular y modificar todas las referencias que sean necesarias de su ontología para que se encuentre al día con respecto a lo que referencia (PURL, 2015).

3.2.2.3.5. Otras ontologías

Se citan aquí otras ontologías encontradas, de gran importancia en el mundo de la Web Semántica, pero que no han sido implementadas en la versión final del estudio:

- *Basic Formal Ontology* (BFO) (http://ifomis.uni-saarland.de/bfo/), creada por el proyecto *Forms of Life* y financiada por la *Volkswagen Foundation*, es una pequeña ontología superior que recupera, analiza e integra ontologías de campos científicos. Gantner (2011) construye su ontología (SONADUS), sobre ella

- *Dublin Core* (http://dublincore.org/documents/2012/06/14/dcmi-terms/?v=elements#), publicada por la iniciativa del mismo nombre, posee una descripción de propiedades y clases básicas muy útiles para la creación de ontologías

Otras ontologías que, en principio, parecían de utilidad pero ya no se encuentran disponibles en internet

- Geographic Information Metadata (ISO 19115): An ontology representing Geographic Information Metadata (http://loki.cae.drexel.edu/~wbs/ontology/)

- Several ISO Geographic Information Ontologies developed with the Protégé-OWL editor. Escritos por Islam, A. S.; Beran, B.; Bermudez, L.; Fellah, S. y Piasecki, M. (http://loki.cae.drexel.edu/~wbs/ontology/list.htm)

- OGC: Ontology for Geography Markup Language (GML3.0) of Open GIS Consortium (OGC). Escrito por: Defne, Z.; Islam, A. S. y Piasecki, M. (http://loki.cae.drexel.edu/~wbs/ontology/ogc-gml.htm)

3.2.2.4. Vocabularios

Según la W3C (2008), en su definición de vocabularios: un vocabulario sirve para integrar datos cuando existen ambigüedades entre dos bases de datos diferentes o cuando es necesario un conocimiento extra para el descubrimiento de nuevas relaciones.

Como ya se ha comentado, se puede comprobar que esta última definición es similar con los conceptos subyacentes poco desarrollados a los que hace referencia Gantner (2011).

Debido a lo específico de la ontología que se pretende implementar, se ha de destacar el esfuerzo empleado para encontrar bibliotecas que fueran de utilidad para el proyecto. Es por esto por lo que se ha decidido crear un vocabulario propio, específico para la base de datos relacional del linaje catastral con la que se deseaba trabajar.

A continuación se citan los resultados que obtuvieron las dos referencias principales encontradas sobre los que se basará el vocabulario creado.

Se las incluye en este apartado debido a su reducido tamaño y la concreción de su universo de discurso.

3.2.2.4.1. Spatiotemporal Ontology for the Administrative Units of Switzerland (SONADUS)

Gantner en su trabajo de fin de master de 2011 enunciaba que la evolución de las unidades administrativas en Suiza había acelerado drásticamente en los últimos 20 años, con continuos cambios de asignación de los terrenos a los municipios. Esto era un problema grave ya que cada municipio es el encargado de gestionar y mantener sus terrenos asignados.

Frente a esta problemática enuncia que la mejor manera de aproximarse al problema es mediante la implementación de una base de datos espaciotemporal y una ontología.

Aprovecha BFO como marco de referencia común para obtener, en su opinión, ventajas en cuestión de interoperabilidad.

Tras el estudio de SONADUS, las tremendas diferencias entre las divisiones administrativas de Suiza y España han llevado a descartar este vocabulario como la base necesaria para el presente trabajo de fin de máster.

3.2.2.4.2. Jurisdictional Domain Ontology (JDO)

Creada por Lacasta et al. en 2014 (Lacasta et al., 2014), describe una ontología para una base de datos relacional que posee datos de la evolución de los dominios jurisdiccionales. Implementa un proceso por el cuál la creación de una base de datos que contenga este tipo de datos sea mucho más fácil. Como ejemplo usa la evolución de las divisiones administrativas de España desde 1830 hasta 2011.

Se ha seleccionado la ontología JDO como base para la definición de la ontología a utilizar en este estudio, ya que, tanto por situación como por contenido, es la que más se aproxima a los datos que se quieren utilizar. Mediante la expansión de este vocabulario se pretende avanzar en la creación de una ontología que abarque todos los matices del catastro español.

3.3. Fuente de datos

Como ya se ha mencionado, los datos son cruciales para la creación de una ontología. De la realización de un buen estudio previo de los datos depende en gran parte el trabajo que realizará posteriormente.

3.3.1. Datos de entrada

Del estudio de Moreno et al. se han obtenido datos de linaje catastral:

- Información sobre las parcelas (archivo *shapefile*, 4 archivos: .dbf, .prj, .shp, .shx)
- Relaciones de linaje (archivo de datos, un archivo: .dbf)

Antes de poder manipular los datos será necesario asignar el sistema de referencia y, posteriormente, sistema de gestión PostreSQL.

Los datos cargados dan información sobre 11.246 parcelas, que ocupan una superficie de 5.376,1302 ha, en cinco municipios diferentes de dos provincias (y comunidades autónomas) diferentes.

El linaje, en cambio, trata de tan solo 204 modificaciones realizadas en un total de 95 parcelas de origen (o parcelas padre) y 77 de resultado (parcelas hijo). Estas modificaciones no se realizan sobre la totalidad de parcelas de la base de datos, sino que se aplican a una pequeña muestra de las mismas: 210, que suman un área de 259,0581 ha.

3.3.2. PostgreSQL

Es un sistema de gestión de bases de datos relacionales orientado a objetos. Es libre y está publicado bajo la licencia BSD.

Está dirigido por una comunidad de desarrolladores que trabajan de forma desinteresada y libre, aunque también pueden recibir el apoyo de organizaciones comerciales.

Sus características más destacables son:

- Alta concurrencia, que permite que varios usuarios se conecten a una tabla para consultar y modificar al mismo tiempo
- Variedad de tipos nativos, provee soporte para los tipos de datos más utilizados
- Soporta claves externas (*foreign keys*)

- Disparadores (*triggers*), que realizan acciones tras un evento específico
- Funciones, bloques de código que se ejecutan en el servidor

Debido a la gran expansión de postgresql se estima que su comunidad de usuarios será un gran activo a la hora de realizar el estudio, por tanto se selecciona este sistema de gestión para tratar los datos de origen del presente estudio.

3.3.3. Formato

El formato en el que se encuentran los datos es muy importante ya que de ello dependerán los programas que se podrán utilizar para su manipulación.

En gran medida el formato está condicionado por los programas y las acciones que se desean hacer, pero en otras ocasiones son los formatos los que condicionan el software a utilizar y los trabajos que realizar.

Como se explicará a continuación, gracias a su sencillez y versatilidad se ha decidido exportar los datos de entrada en CSV, para facilitar los procesos de transformación que se les aplicarán posteriormente.

La decisión de guardar la base de datos en RDF ha sido motivada por el uso del software Virtuoso como herramienta final de consulta, ya que dicho programa no acepta bases de datos en otro formato.

3.3.3.1. *Shapefile*

Los archivos *shapefile* son un formato de datos espaciales desarrollados por ESRI para sus Sistemas de Información Geográfica pero actualmente se ha convertido en un estándar de facto.

Es un formato vectorial de almacenamiento que guarda la localización de elementos geográficos y sus atributos, pero no guarda información topológica. Es un formato "multiarchivo" y llevan, obligatoriamente tres tipos de archivos conjuntamente:

- SHP, que contiene la geometría en si misma
- SHX, un índice posicional que permite la búsqueda en la geometría de una manera más rápida

DBF (*DataBase File*), contiene las tablas con toda la información requerida Opcionalmente puede llevar:

PRJ, un texto que define el sistema de coordenadas y la proyección.

Los datos de origen provienen de ArcGIS, por lo que se trabajará con un *shapefile*.

3.3.3.2. CSV

Comma Separated Values, es un tipo de documento en formato abierto. Es el más sencillo para la representación de los datos en forma de tabla ya que las columnas se separan por comas (o punto y coma para lugares donde el separador decimal es una coma) y las filas por saltos de línea. Los campos que posean una coma, un salto de línea o un a comilla doble deben de encontrarse entre comillas dobles.

Al ser el formato más sencillo de base de datos también es el que más programas soportan, por lo que es un formato ideal para la realización de un estudio donde se pretende manipular los datos de muy diferentes maneras.

Otros formatos como el archivo de texto o archivos obtenidos de hojas de cálculo no serían tan funcionales y no tendrían tantas ventajas.

3.3.3.3. RDF

Siglas en inglés de Marco de Descripción de Recursos (*Resource Description Framework*). Es una familia de especificaciones de la World Wide Web Consortium, originalmente diseñado como un modelo de datos para metadatos.

Ha llegado a ser usado como un método general para la descripción conceptual o modelado de la información. Se implementa en los recursos web, utilizando variedad de notaciones de sintaxis y formatos de datos (W3C, 2014).

3.3.3.4. OWL

Es un lenguaje de etiquetas semántico utilizado para publicar y compartir ontologías en World Wide Web.

OWL (*Web Ontology* Language) se desarrolla como un vocabulario de extensión de

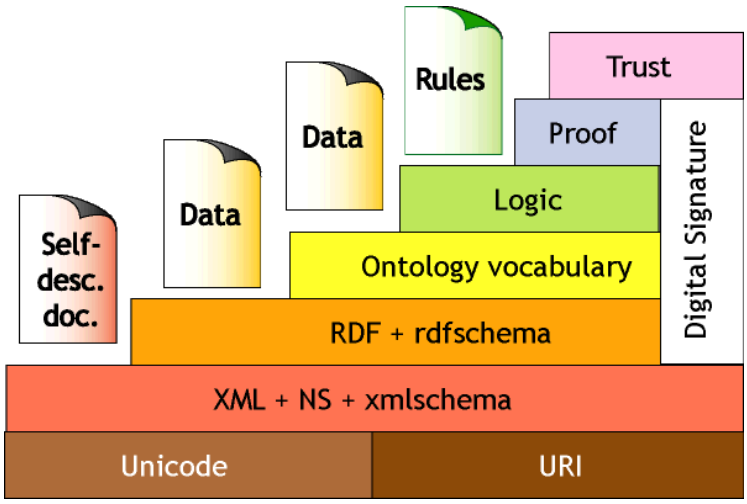

Figura 7. Estructura de OWL

RDF deriva de DAML+OIL *Web Ontology Language*.

La primera recomendación se publicó el 10 de febrero de 2004 y, actualmente, se encuentra por la segunda, publicada el 27 de marzo de 2009.

Esta segunda revisión proporciona clases, propiedades, individuos y valores de datos guardados como documentos de Web Semántica. Pueden ser utilizados junto con información escrita en RDF.

3.3.4. Linaje en el catastro de España

Los datos catastrales obtenidos de la Sede Electrónica del Catastro (SEC), no incluyen información de su genealogía, por lo que la gestión histórica de las parcelas es muy limitada.

Gracias al trabajo de Moreno et al. (2014), se consigue, mediante una serie de algoritmos implementados como consultas en un gestor de base de datos espacial, conseguir con un grado satisfactorio de eficacia las relaciones de agregación y segregación de las parcelas.

La metodología se ha restringido sólo a este tipo de relaciones (agregación y segregación) debido a que son las más extendidas en el catastro español. Por otro lado se comprobó que otros tipos de relaciones tenían diferentes problemas a causa de los datos de origen.

Este estudio propone también un prototipo de estructura relacional para el almacenamiento y la gestión de datos de manera pública.

Con el presente trabajo de fin de master se pretende completar la funcionalidad de esta base de datos otorgándole los beneficios que reporta la Web Semántica mediante la implementación de una ontología para los datos de linaje catastral.

3.3.5. Base de datos espacio-temporal

Como su nombre indica es una base de datos que maneja, de manera simultánea, información espacial e información temporal.

Se pueden considerar una extensión de las bases de datos espaciales. Gestionan el cambio de la geometría a lo largo del tiempo.

No existen demasiados ejemplos de bases de datos relacionales con componente temporal debido a la gran complejidad que el tiempo aporta a las bases de datos. Muchos de ellos simplemente guardan la característica del tiempo como un dato en formato numérico o de texto sin implementar funciones de consulta temporal.

3.4. Lenguajes de consulta

Lenguaje informático que sirve para obtener datos de bases de datos o sistemas de información.

Existe gran variedad de lenguajes de consulta, que puede ser estándar o específicamente creado para un programa.

Algunos ejemplos son: *Common Query Language* (CQL); *Object Query Language* (OQL)… Los más extendidos poseen una mayor comunidad de usuarios y mayor facilidad para su estudio e implementación. Por ello serán los que se utilicen durante el estudio. Se explican con mayor detalle a continuación.

3.4.1. SQL

Es un lenguaje estructurado de consulta que sirve para acceder a bases de datos relacionales y permite especificar diversos tipos de operaciones con ellas. Explota su flexibilidad y potencia, permitiendo así gran variedad de operaciones.

Una de sus características es el manejo del álgebra y el cálculo relacional que permiten efectuar consultas con el fin de recuperar, de manera sencilla, información de bases de datos, así como hacer cambios en ellas.

Los sistemas de gestión de bases de datos más utilizados que soportan SQL son: PostgreSQL, Oracle, MySQL, Firebird, Microsoft SQL Server…

3.4.2. SPARQL

SPARQL guarda cierta similitud con el lenguaje de consulta a bases de datos SQL, pero se diferencia en que las sentencias que utiliza para la consulta se basan en tripletas de conceptos Sujeto-Predicado-Objeto.

Asume que no se puede garantizar que la información descubierta sea completa, es decir, si algo no aparece, es desconocido, no falso.

Permite consultar a múltiples colecciones para posibilitar la integración de la información. En la actualidad existe la versión SPARQL 1.0 y en borrador la SPARQL 1.1, la cual no está implementada en todos los servidores de datos RDF.

Se puede usar de dos modos, uno de ellos a través de consultas vía API (*Application Programming Interface*) de repositorios RDF (*Jena2, Sesame, OpenLinkVirtuoso, Pellet, Protégé…*) o a través de consulta web a un punto de entrada (*endpoint*) SPARQL. En el presente estudio se usará, principalmente, el primer modo.

3.5. Manipulación de datos

A lo largo de las fases de trabajo ha sido necesaria la continua manipulación y modificación de los datos. Como se podrá apreciar más adelante, cada uno de los pasos realizados en el presente trabajo de fin de master ha requerido de uno o varios programas para llevar a cabo las tareas asignadas.

En todo momento se han buscado herramientas de software libre y código abierto, buscando en todo momento el objetivo de interoperabilidad que posee este estudio.

3.5.1. *Geokettle*

Es una versión de *Pentaho Data Integration* (Kettle) con capacidad de tratamiento de datos espaciales.

Software del tipo ETL (*Extract, Transform, Load*: Extracción, Transformación, Carga) orientado al uso de metadatos y con funcionalidades espaciales. Está dedicada a la integración de diversos orígenes de datos para la construcción y actualización de bases de datos espaciales y almacenes de datos espaciales.

Permite la extracción de información de los gestores de bases de datos más extendidos, de archivos XML, de servicios OGC Web, de formatos de datos geoespaciales…

Sus labores principales de transformación pasan por: corregir errores, realizar limpieza de datos, modificar su estructura, hacer cumplir los estándares, cargar sistemas de referencia…

Por último se puede realizar la carga de datos a un DBMS, a un archivo SIG, un servicio web geoespacial…

Es de gran utilidad cuando se pretenden automatizar procesos complejos y repetitivos sin necesidad de generar código de programación para ello (OSGeo, 2010).

Se le puede comparar con FME en el ámbito geoespacial, una herramienta de pago que realiza ETL. Geokettle es estable, rápido, cumple con los estándares, tiene cientos de funciones y lee y escribe en diversos formatos de ficheros, servicios y DBMS.

3.5.2. *pgAdmin III*

pgAdmin III es una herramienta de código abierto que sirve para la administración de bases de datos *PostgreSQL* y derivados (*EnterpriseDB*, *Postgres Plus*, *Advanced Server* y *Greenplum Database*).

Ofrece:

- interfaz gráfica
- herramienta de consulta SQL (con un *EXPLAIN* gráfico)
- Editor de código procedural
- Agente de planificación SQL/Shell/batch
- Administración de Slony-I: sistema de replicación para postgresql

La interfaz gráfica soporta todas las características de PostgreSQL y hace simple la administración.

Su interoperabilidad es muy grande ya que está disponible en más de una docena de lenguajes y para varios sistemas operativos, incluyendo Microsoft Windows, Linux, FreeBSD, Mac OSX y Solaris. Soporta versiones de servidores 7.3 y superiores.

Otros programas de administración de base de datos pueden ser DbBisualizer, Database Browser, Navicat… Todas ellas gratuitas, pero todavía en fase de desarrollo, que no llegan a implementar tantas herramientas como pgAdmin.

La decisión para utilizar este programa para el estudio ha sido debido a su capacidad operacional (pudiendo escribir simples consultas SQL o desarrollar bases de datos complejas), su expansión en el mundo profesional, su condición gratuita y su interoperabilidad. Que concuerda a la perfección con los requerimientos del presente estudio.

3.5.3. *PROTÉGÉ*

Se ha podido comprobar que actualmente existen lenguajes estándar y una gran variedad de herramientas (tanto comerciales como de código libre) para crear y trabajar con ontologías.

Desarrollando por la universidad de Stanford, en colaboración con la universidad de Mánchester, *Protégé* es un editor de ontologías libre de código abierto y un sistema de adquisición de conocimiento. Es un *framework* pare el cual otros proyectos sugieren *plugins*.

La aplicación está creada en *Java* y usa *Swing* para crear su interfaz.

Tiene más de 100.000 usuarios registrados, por lo que está respaldado por una comunidad científica muy importante.

Otros programas similares a *Protégé* son: SWOOP, Pellet y Virtuoso.

Se ha seleccionado *Protégé* ya que prácticamente la totalidad de ontologías que se han encontrado y manipulado habían sido desarrolladas en este programa y, por tanto, haría más efectivo el estudio de las mismas.

3.5.4. *Eclipse*

Al igual que *Protégé*, es un entorno de trabajo.

Es un programa informático compuesto por un conjunto de herramientas de programación de código abierto, multiplataforma, para desarrollar aplicaciones. Está desarrollado en la actualidad por la Fundación Eclipse, una organización sin ánimo de lucro que fomenta una comunidad de código abierto y un conjunto de productos complementarios, capacidades y servicios. En él se pueden desarrollar aplicaciones en Java, Android, C++… También posee una comunidad de usuarios que extiende continuamente las áreas de aplicación de eclipse.

Las alternativas gratuitas a *Eclipse* más populares son: Vim, NetBeans, Microsoft Visual Studio…

Eclipse ha sido la herramienta elegida por tener un entorno con una comunidad de usuarios más extensa ya que el trabajo que se pretende hacer con él es sencillo y puede apoyarse perfectamente en la comunidad.

3.5.5. *Virtuoso*

Virtuoso, en su versión 7.10.3207, es un software intermedio y un motor de bases de datos híbrido que combina la funcionalidad de las RDBMS, ORDBMS, la base de datos virtual, el RDF, el XML, una aplicación web y un servidor de ficheros en un solo sistema. Se le considera un "servidor universal" que permite hacer múltiples procesos que implementa múltiples protocolos.

También se le conoce como *OpenLink Virtuoso* debido a su condición de programa libre y gratuito

Utiliza SPARQL para hacer consultas semánticas a las bases de datos cargadas en su servidor y soporta consultas geoespaciales.

Programas similares que pueden ser utilizados para consulta de bases de datos con SPARQL son *Pellet*, *Protégé*, *Open Refine*…

El único punto negativo que se encontró en *Virtuoso* en el momento de la fase de estudio fue la imposición del formato RDF para el SGBDR en lugar del OWL, formato aprobado por la W3C.

Más tarde, en la fase de consulta, se descubrió que *Virtuoso* contaba con otro punto negativo: las consultas espaciales soportadas sólo permitían la búsqueda de la interacción entre puntos y geometrías, no entre dos geometrías.

Gracias a su gran potencia y posibilidades para aplicaciones futuras se ha decidido utilizar *Virtuoso* al comienzo del estudio.

4. Metodología

En este apartado se describirán, de manera pormenorizada, los procesos seguidos durante el estudio.

A modo de resumen, se presenta la Figura 8: diagrama de flujo que divide el trabajo de fin de master en las fases consideradas, enunciando los trabajos realizados en cada una de ellas.

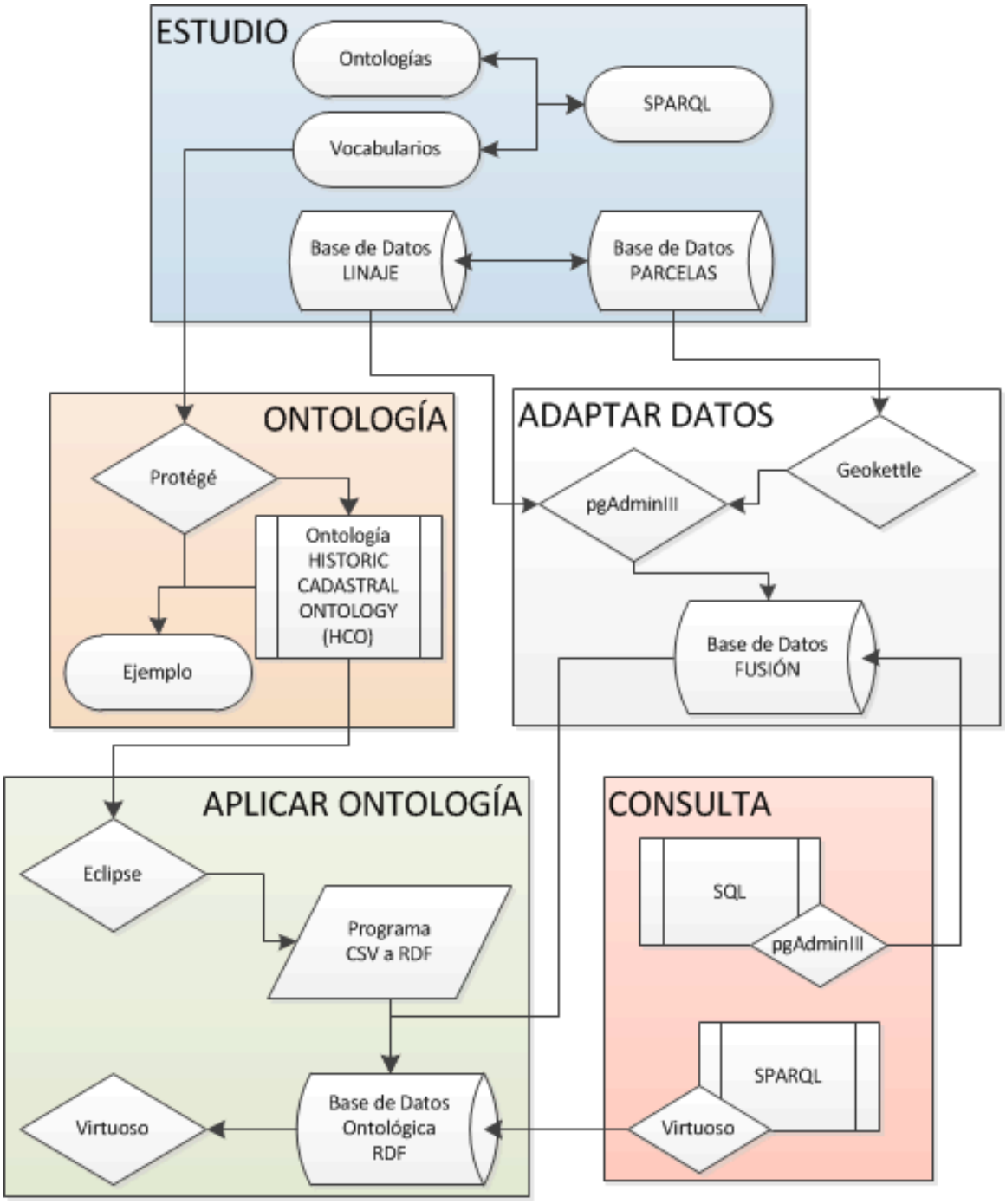

Figura 8. Diagrama de flujo del estudio

La fase de "estudio", como se viene tratando en el presente informe, ha de ser una de las fases más extensas en cuanto a tiempo de trabajo se refiere. No solo engloba el estudio de las ontologías, vocabularios y el lenguaje de consulta SPARQL, sino que también se ha de realizar un estudio en profundidad de los datos de linaje catastral.

Esta fase finalizará al adquirir los conocimientos necesarios para la toma de decisiones pertinentes para la implementación (construcción, almacenado y explotación) de la ontología.

Mediante el programa *Protégé*, en la fase "ontología" se creará el vocabulario teniendo en cuenta los requerimientos de la base de datos de linaje catastral. Tras la creación de la ontología se creará, a mano, un pequeño ejemplo para hacer pruebas básicas de funcionamiento.

La base de datos de linaje catastral es muy extensa y compleja, por lo que para poder aplicarle la ontología será necesario, como paso previo a la aplicación de la ontología "adaptar datos" de las parcelas y de linaje catastral para que el proceso de "aplicar ontología" sea mucho más rápido y fácil.

En este momento se tiene todo lo necesario para proceder a aplicar la ontología a la base de datos relacional, por lo que se procederá a "aplicar ontología". Para ello primero se usará un programa explícitamente creado para este estudio que automatice la tarea de transformar la tabla del SGBDR en tripletas sujeto-predicado-objeto en formato RDF. Las tripletas se cargarán a *Virtuoso* y se comprobará que los datos han sido correctamente guardados.

Finalmente la fase "consulta" hace comparaciones entre consultas SQL y SPARQL a los mismos datos para así ofrecer unos resultados y conclusiones al estudio.

4.1. Estudio

En esta fase se ha de tener en cuenta que para este trabajo de fin de máster son necesarios los conocimientos adquiridos sobre SGBDR, SQL, ontologías y SPARQL en las asignaturas de *Bases de Datos Espaciales*, *Infraestructura de Datos Espaciales* y *Programación Web*; y que dichos conocimientos serán prácticamente suficientes para los SGBDR y el lenguaje SQL, pero que será necesario ampliar los conocimientos de ontologías y SPARQL para una correcta y completa realización del estudio.

4.1.1. Bases de datos

El primer paso para poder hacer una definición del ser, es conocer el propio ser que se pretende definir, es decir, se ha de tener un conocimiento profundo de todos los matices de la base de datos relacional de linaje catastral antes de realizar la ontología.

Las dos tablas de datos de origen han sido cargadas a un sistema PostgreSQL mediante la herramienta de trabajo pgAdmin para, mediante consultas SQL y observación directa de las tablas, poder analizar los datos que ellas contienen.

Una vez cargadas las tablas se procedió a la creación de índices y restricciones para que las búsquedas fueran mucho más rápidas y cómodas durante el proceso de estudio de la base de datos.

Las Claves Primarias (PK en sus siglas en inglés) son restricciones que identifican de manera única cada fila de la tabla. El campo elegido debe no puede repetir su valor ni ser nulo, por tanto para las PK han sido los identificadores de cada tabla (números correlativos que están asignados a cada registro de la tabla).

Debido a esto primero se comprobó si los datos que vamos a usar son únicos en la base de datos:

```
SELECT ninterno, COUNT(ninterno)
      FROM parcelas GROUP BY ninterno HAVING COUNT(ninterno)>1;
```

Si la consulta da 0 resultados quiere decir que el dato no se repite, en caso de haber algún dato repetido se mostrará el dato con el número de repeticiones a su lado.

Mediante estas consultas se crearon las PK:

- En “parcelas”:

```
ALTER TABLE _parcelas
      ADD CONSTRAINT "_ID_pkey" PRIMARY KEY("ID");
```

- Y en los datos de “linaje”:

```
ALTER TABLE _linaje
      ADD CONSTRAINT "_ID_PK" PRIMARY KEY(id);
```

Permiten habitualmente una búsqueda de datos más rápida y una mejor relación entre datos de diferentes tablas.

Los índices en cambio sí permiten que los valores no sean únicos y que existan casos en los que sea nulo.

Se creó un índice para la tabla “parcelas”:

```
CREATE INDEX _refcat_idx
  ON _parcelas
  USING btree
  (refcat COLLATE pg_catalog."default" text_pattern_ops);
```

Y cinco índices para “linaje” en los datos de: *identificador*, *referencia catastral hijo*, *referencia catastral padre*, *número interno hijo* y *número interno padre*.

Gracias a la construcción de estos índices se puede buscar por patrón, sin necesidad de escribir todo el texto para obtener resultados a la consulta.

Un árbol-B es un tipo de organización jerárquica de los datos que permite una búsqueda rápida y efectiva.

Mejoran el rendimiento de la base de datos. Permite al servidor de la base de datos encontrar y mostrar celdas específicas de una manera mucho más rápida, pero cargan un mayor peso a la base de datos, por lo que se deben de usar con cuidado (PostgreSQL, 2015).

El archivo correspondiente a parcelas es un *shapefile* en el que se puede encontrar la información espacial georreferenciada por lo que, además, se creó un índice espacial para futuras búsquedas con la siguiente consulta SQL:

```
CREATE INDEX the_geom_idx ON _parcelas USING GIST(the_geom);
```

Una vez preparada la base de datos se procedió a su estudio. El primer paso fue familiarizarse con los datos que se presentaban en las tablas. “parcelas” contaba con 25 campos de los que destacan:

- referencia catastral
- número interno (que sirve de identificación a nivel del estudio)
- área
- geometría
- fechas de validez

“linaje” presentaba 14 campos, de los que los más importantes son:

- referencias catastrales de parcelas padre e hijo
- áreas de padre e hijo
- cambio que lleva del padre al hijo
- fechas de inicio y fin de ambos

El dato básico del estudio es el “cambio” que se indica en la tabla linaje, por ello, mediante el apoyo en la documentación que acompañaba a los datos, se tuvieron en consideración todas las modificaciones posibles.

Tabla 1. Descripción de los tipos de cambio que puede presentar una parcela

Cambio	Descripción
0	*Sin cambios*
1	*Rectificación* de límites
100	*Aparición* (a partir de terrenos públicos, otras causas…)
101	*Aparición* por *Agrupación* de otras parcelas
102	*Aparición* por *Agregación*
103	*Aparición* por *División* de otra parcela
104	*Aparición* por *Segregación*
2	*Cambios temáticos*
900	*Desaparición* (por integración a terreno público…)
901	*Desaparición* por *Unión*
902	*Desaparición* por *Agregación*
903	*Desaparición* por *División*
904	*Desaparición* por *Segregación*

La búsqueda de la ontología para implementar se basó, posteriormente, en la información otorgada por esta tabla.

4.1.2. Ontologías y vocabularios

Se comenzó estudiando a fondo los apuntes obtenidos durante el estudio del máster para así afianzar la base teórica y localizar un punto de partida para comenzar la implementación de la ontología: los buscadores de ontologías.

Como apoyo a los datos ontológicos que se iban a buscar se hizo una búsqueda de trabajos anteriores similares a lo que se pretendía en el presente trabajo a modo de referencia. Se consideraron de especial interés aquellos estudios realizados en el ámbito español ya que las divisiones administrativas cambian en cada país. En este punto se encontraron las dos referencias principales que se citan en el estudio:

- *A Spatiotemporal Ontology for the Administrative units of Switzerland*, tesis de máster escrita por Felix Gantner en 2011; crea SONADUS (*Spatiotemporal Ontology for the Administrative Units of Switzerland*), basado en BFO (*Basic Formal Ontology*)

- *Population of a spatio-temporal knowledge base for jurisdictional domains*, escrito por un equipo formado por miembros del departamento de ciencias de la información y sistemas de ingeniería de la Universidad de Zaragoza y del centro de investigaciones unidas de la comisión europea (citado como “Lacasta et al., 2014”), donde crean una ontología completamente nueva: JDO (*Jurisdictional Domain Ontology*)

Debido al objetivo de interoperabilidad que se busca con las ontologías se decidió escribir la del presente estudio en inglés, lengua habitual utilizada en este campo. Las descripciones en castellano se podrán agregar como una característica más de cada sujeto.

Tras descartar los buscadores de ontologías como herramienta para el estudio se pasó al estudio de los listados de ontologías, por ser considerados como estándares en sus correspondientes ámbitos y aportando, por tanto, la interoperabilidad deseada.

Finalmente se comprobó que no existía ninguna ontología que se ajustara al ámbito catastral español ni que otorgara una solución adaptable a los datos de linaje catastral por lo que se tomó la decisión de crear un vocabulario específico para este SGBDR basándose en el marco otorgado por el trabajo de Lacasta et al. en 2014: JDO.

4.1.2.1. Buscadores de ontologías

Tras seleccionar el sujeto de la base de datos (parcela catastral) y las características del mismo que se querían describir (referencia catastral, área…) y sobre todo las relaciones posibles de cambio (Aparición, Desaparición, Rectificación de Límites…) y tras dejar clara la definición que se pretendía implementar, se estaba en disposición de comenzar la búsqueda de una ontología afín.

4.1.2.1.1. Linked Open Vocabularies

El buscador LOV, demostró pronto ser el más útil en el campo de los motores de búsqueda de ontologías, ya que además de la gran cantidad de ontologías indexadas, éstas eran de carácter científico, lo cual hacía que la búsqueda de la más adecuada fuera mínimo.

Otra de las ventajas observadas en LOV es la claridad y buen diseño de la página de resultados, que permite al usuario hacerse una idea de todas las características de los resultados obtenidos. Aparte de mostrar el resultado, su URI y sus etiquetas, muestra

Figura 9. Ejemplo de resultado en LOV

una serie de opciones para detallar la búsqueda: Tipo, Etiquetas, Vocabulario.

Gracias a este buscador se encontraron ontologías de interés, pero no fue posible encontrar definiciones para todos los elementos de la ontología.

A continuación se citan los elementos encontrados que más se adecúan a la ontología buscada, pero que terminaron siendo descartadas:

- Aparición: http://purl.org/dc/terms/created (*created*)
  Característica que podría ser ambigua
- Agregación:
  http://rdvocab.info/RDARelationshipsWEMI/mergedWithToForm (*merge*)
- División: http://bibframe.org/vpcab/splitInto (*split into*)
- Cambios temáticos: http://voag.linkedmodel.org/voag#ChangeType (*change type*)
- Agrupación: http://www.w3.org/ns/prov#Association (*association*)
  http://www.w3.org/2002/07/owl#unionOf (*union of*)
- Segregación: http://www.w3.org/2002/07/owl#disjointWith (*disjoint with*)
  http://data.ordnancesurvey.co.u/ontology/spatialrelations/disjoint (*disjoint*)

Varias ontologías podrían ser aptas para describir un mismo término, por lo que, por motivos de funcionalidad, se buscó la que más términos fuera capaz de describir, el resultado fue un máximo de dos términos tras una semana de trabajo.

Se estimó que los motores de búsqueda no eran el método ideal para este trabajo debido a las características del mismo y a los pocos resultados obtenidos comparados con la cantidad de trabajo dedicado.

4.1.2.2. Bibliotecas de ontologías

Tras descartar los motores de búsqueda de ontologías se pasó al estudio de los listados de ontologías que se encuentran disponibles en la red.

Se comprobó rápidamente que la utilidad de las mismas era mayor que la de los buscadores para el estudio ya que agrupaba ontologías completas de manera temática. Esta división temática también evidenciaba un nivel de especificación mayor que el encontrado en los resultados de los buscadores.

De esta manera el buscador pasó a ser el usuario, capaz de mayor adaptabilidad que el algoritmo creado para cualquier buscador. Con la visión general de la ontología

que se quería implementar se buscó a través de las bibliotecas aquellas que fueran de mayor utilidad para el fin del estudio.

Se consultaron varios listados de ontologías siendo los más completos y útiles los descritos anteriormente en el estado del arte.

4.1.2.2.1. Semantic Web

Semantic Web es un listado de gran interés debido a las ontologías de carácter general que describe. Las ontologías se usan para la definición de los caracteres básicos de la Web Semántica, debido a ello no es de utilidad para un proyecto que busca unas definiciones tan concretas como este.

4.1.2.2.2. COLORE

En el listado COLORE se encuentra la mayor cantidad de ontologías vista en ningún listado, su objetivo es crear un listado de ontologías básicas para servir de conceptos primitivos para la Web Semántica. Ejemplos de ontologías encontrados en este listado son: *cyclic_geometry*; *incidence_geometry*; *multigeometry* y *ordered_geometry*; estas ontologías son demasiado básicas y ninguna de ellas se aproximaba lo suficiente como para poder ser adaptable a la información de linaje catastral.

4.1.2.2.3. DARPA Agent Markup Language

La biblioteca DAML contiene una extensa colección de ontologías de temas muy variados, el ámbito geográfico no es una excepción para este listado que pretende facilitar la implementación de la Web Semántica.

Se encontraron muchas ontologías de ámbito geográfico, las que más interés suscitaron fueron: *geography*; *geofile-ont* y *geonames-ont*; del estudio de éstas ontologías se deduce que existen muchos términos confusos y no muy bien agrupados dentro de las mismas por lo que no se estimó adecuado su uso.

4.1.2.2.4. NeOn

El listado NeOn ofrece una serie de ontologías con un rigor científico muy grande y de gran interés.

En ella se encontraron dos ontologías relacionadas con el ámbito geográfico de gran precisión, éstas fueron las candidatas más próximas a la implementación HCO que se encontraron:

- *Geopolitical Entities*, su elemento principal es el área y contempla las maneras en las que actúan entre ellas, no su interrelación, su herencia, ni su valor temporal

- *FAO Geopolitical Ontology*, centrada en las divisiones administrativas y las conexiones entre ellas no tiene el nivel de detalle suficiente como para poder definir una parcela catastral

4.1.2.3. SPARQL

Como ya se ha mencionado el SPARQL es el lenguaje de consulta seleccionado para manipular la base de datos semántica y comprobar su funcionalidad, resultados y beneficios. Es por ello que se le dedicó especial atención al aprendizaje y desarrollo de las habilidades pertinentes para el uso de este lenguaje.

Complementariamente a lo estudiado en el transcurso del máster se buscaron una serie de manuales y ejemplos de bases de datos relacionales en diferentes formatos para la experimentación y profundización en el lenguaje de consulta.

La W3C es la organización que definió la especificación de SPARQL para la consulta de datos en RDF, por lo que, primeramente, se buscó su sintaxis y semántica en su portal, encontrando la siguiente recomendación del 15 de enero de 2008: http://www.w3.org/TR/rdf-sparql-query/

Esta recomendación es un completo manual, en inglés, que permite a cualquier usuario con conocimientos mínimos de bases de datos y de lenguajes de consulta, aplicar sin problemas consultas SPARQL a bases de datos semánticas.

Sin embargo esta guía también tiene un apartado para hacer consultas sencillas, información muy útil para usuarios que sólo necesitan SPARQL en sus funciones básicas.

El resto del manual abarca la sintaxis del lenguaje, los patrones gráficos de consulta, tanto sencillos como complejos y con subconsultas.

Apache Jena creó un rápido curso de iniciación a SPARQL, mucho menos extenso y más fácil y sencillo de entender que el anterior (Apache Jena, 2011).

Posee gran cantidad de ejemplos que muestran los conceptos que se van mostrando a lo largo de este pequeño curso. Por otro lado también hace referencia a otros documentos que han creado para usuarios más avanzados.

El "grupo de tareas de las mejores prácticas para RDF/OWL" de los estándares de información de la biodiversidad (*RDF/OWL Best Practices Task Group* como parte de *Biodiversity Information Standards* o TDWG) se encarga de traducir los vocabularios de la TDWG para ser usados como clases de RDF. Dentro de ese marco de trabajo el grupo ha creado un completo manual de RDF que incluye un apartado que enseña el uso del SPARQL. Este manual puede ser encontrado aquí: https://code.google.com/p/tdwg-rdf/wiki/Beginners6SPARQL.

Dentro de la búsqueda de manuales y ejemplos se dio con una página bastante interesante que enuncia una serie de consultas básicas de ejemplo que son de utilidad para el usuario principiante: https://code.google.com/p/void-impl/wiki/SPARQLQueriesForStatistics.

Por último la página http://sparql.org resulta de una utilidad enorme cuando se dan los primeros pasos en la investigación del SPARQL, ya que pone a disposición del usuario dos aplicaciones web con ejemplos sencillos de bases de datos semánticas donde comenzar a hacer consultas y 5 validadores de consultas, de gran utilidad una vez se salga de su entorno controlado.

Finalmente, en caso de que el usuario tenga alguna dura mientras utiliza su página, tiene un apartado con hiperenlaces a las páginas de la W3C más interesantes en cuestión de SPARQL.

4.1.2.4. Herramientas

La fase de estudio también incluyo, como es necesario, la búsqueda de aplicaciones o herramientas que son necesarias para el objetivo del estudio.

En primer lugar se estudió la manipulación de datos en pgAdmin que, como ya se comentó, fue la aplicación elegida para la manipulación del SGBDR. La necesidad de profundizar en el uso de este programa se hizo más evidente según se fue avanzando en el trabajo de fin de master.

La propia aplicación de pgAdmin tiene en su portal en línea un extenso documento que muestra y enseña a manejar todas las funciones que ofrece al usuario: http://www.pgadmin.org/docs/1.20/index.html.

También fue de gran utilidad en la fase de adaptación de la base de datos: http://www.postgresql.org/docs/9.1/static/index.html.

Después de esto se buscaron las herramientas software necesarias para la implementación del vocabulario que se pretendía crear.

Se hizo hincapié en el programa de creación y manipulación de ontologías Protégé. Su página principal (http://protege.stanford.edu) además de ofrecer la descarga gratuita de sus productos, enlaza a una enorme comunidad de usuarios que aportan un gran valor añadido.

Tras la creación del vocabulario el siguiente paso consiste en aplicarlo al SGBDR, por lo que era necesaria una herramienta de transformación.

La información que se encontró al respecto indicaba que se debía de transformar la base de datos a RDF. Sobre el cambio de base de datos relacional a RDF (RDB2RDF en sus siglas en inglés) se encontró mucha literatura de la cual se destaca:

- El grupo de trabajo de la W3C RDB2RDF creado en 2007: http://www.w3.org/2001/sw/rdb2rdf/
- La recomendación creada por dicho grupo en septiembre de 2012: http://www.w3.org/TR/r2rml/
- Casos de uso y requerimientos para el paso de bases de datos de mapas a RDF: http://www.w3.org/TR/rdb2rdf-ucr/
- Un compendio de trabajos sobre RDB2RDF: http://semanticweb.memect.com/?tag=rdb2rdf
- La encuesta creada por el grupo Incubator de la W3C sobre las aproximaciones para el paso de bases de datos relacionales a RDF (Incubator, 2009)

El problema del paso de RDB a RDF, como se pudo comprobar es muy recurrente, ya que se encontró con relativa facilidad varios listados de programas que decían cumplir ese cometido:

- Lista en la que separan los transformadores por el tipo de archivo que pasan a RDF: http://www.w3.org/wiki/ConverterToRdf
- Lista especifica de RDB2RDF: http://www.w3.org/2001/sw/wiki/RDB2RDF

En estos listados se mencionan diferentes aplicaciones con funcionalidad y cualidades muy diferentes que abarcan un abanico tremendamente amplio de posibles trabajos, a modo de ejemplo se citan los más completos:

- http://d2rq.org/ que soporta SPARQL y tiene una herramienta de mapeado
- http://triplify.org/Overview posee una base de datos de 160 Gb procedente de OpenStreetMap, pero no soporta SPARQL
- https://github.com/antidot/db2triples iniciativa de código libre, almacenada en un hub de datos, está basado en un servidor maven.apache

Sin embargo las herramientas más citadas para el paso de una base de datos relacional a RDF son Virtuoso y Refine.

Por comodidad y sencillez primero se probó la funcionalidad de la herramienta de Google Refine. Al cabo de unas pocas pruebas se pudo comprobar que la extensión había cambiado su contenido y ahora sólo sirve para cargar bases de datos metadatados y hacer consultas a los mismos, por lo que quedó descartado para el estudio.

Figura 10. Página inicial de OpenLink Virtuoso

Finalmente se procedió al estudio de Virtuoso, herramienta completa y compleja que, entre otros muchos aspectos hace el paso de RDB a RDF.

Virtuoso es una aplicación tan compleja que necesita de todo un manual de instalación y otro de uso:

- Manual de instalación: http://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/
- Tutoriales: http://virtuoso.openlinksw.com/tutorials/

Tras el estudio de la aplicación se determinó que no funcionaba de la manera que era necesaria para el proyecto y que, por tanto, no se utilizaría para el paso del SGBDR a RDF. Sin embargo, se apreció el gran potencial del programa y se tomó la decisión de usarlo para realizar las consultas SPARQL debido a las posibilidades que ofrecía para desarrollos posteriores del estudio como servidor de los datos.

4.2. Ontología

Una vez terminado el estudio de la base de datos de linaje catastral, de las ontologías, los vocabularios y el SPARQL, se tenían los conocimientos suficientes para la toma de decisiones sobre la definición e implementación de la ontología.

Se estimó que la mejor manera de proceder para implementar una ontología que se ajustara a la base de datos con información de linaje catastral era crear un vocabulario propio, basado en JDO, que sea abierto, libre y en RDF para que pueda ser ampliado en la medida que sea necesario para la base de datos de linajes de Catastro.

A continuación se describen las clases, objetos y propiedades descritas en Protégé que se crearon para la ontología de este trabajo de fin de master.

4.2.1. *Historic Cadastral Ontology* (HCO)

Este vocabulario es una adaptación del creado por Lacasta et al. (2014); por lo que el primer paso para la creación del mismo es cargar una copia de su vocabulario en el programa Protégé.

JDO, además de la base de las divisiones administrativas, aporta al vocabulario todas las definiciones básicas que son necesarias para construirlo: descripción ("description" definido por purl.org), formato ("format", purl.org), identificador ("identifier", purl.org), comentario ("comment", w3c.org/rdfschema)… algunas de estas definiciones también las aporta Protégé por defecto. También fue necesaria la inclusión de la descripción de geometría ("geometry") dada por w3c.org/2003/01/geo/wgs84_pos.

Al nuevo vocabulario se le asignó el nombre de "Historic Cadastral Ontology" por considerarse suficiente para describir el objeto del vocabulario y para poder nombrarlo de una manera más cómoda por sus siglas: HCO.

Se añadió dentro de la clase *information-object* la subclase *cadastral-object* y dentro de ésta la "mínima división del territorio": ***parcel***.

Las "propiedades del objeto" son las relaciones entre parcelas, es decir, todas las modificaciones que contempla el modelo de Moreno et al. (2014) para el linaje entre padres e hijos. Las variaciones que relacionan las parcelas se agruparon en *predecessor* y *succesor* en función de la modificación que defina. Aquellas que no incumben a más de una parcela se han guardado por separado.

Se definieron dos propiedades independientes (aparición y desaparición) complementarias entre sí y seis propiedades de relación entre parcelas (disolución, fusión, incorporación, rectificación de límites, segregación y cambio temático), también complementarias entre sí. Todas estas propiedades quedan resumidas en la tabla 2

Para finalizar el vocabulario se crearon las propiedades de las parcelas (o *data properties* como figura en Protégé). En esta sección se añadieron todas las características que están incluidas en la base de datos relacional de linaje catastral y que ya se habían estudiado con anterioridad.

Fue necesaria la creación de todas las características ya que JDO no consideraba algo tan concreto como una parcela como objeto del vocabulario. Se crearon nueve características:

Tabla 2. Propiedades de objeto de HCO

	appearance
	disappearance
predecessor	*dissolved-into*
	fused-into
	incorporated-into
	rectificates-limits-into
	segregates
	tematically-changes-into
successor	*dissolves*
	fuses
	incorpores
	limit-rectification
	segregated-from
	tematic-change

- *area*, en metros cuadrados
- *change*, cambio que ha sufrido la parcela
- *delegation*, concreta el área donde se localiza el objeto catastral
- *initial-instant*, indica el momento en el que empieza la validez del objeto catastral
- *final-instant*¸ indica el fin de la validez de un objeto catastral
- *geometry*, forma del recurso
- *identifier*, referencia catastral
- *map*, número del mapa donde está localizada la parcela

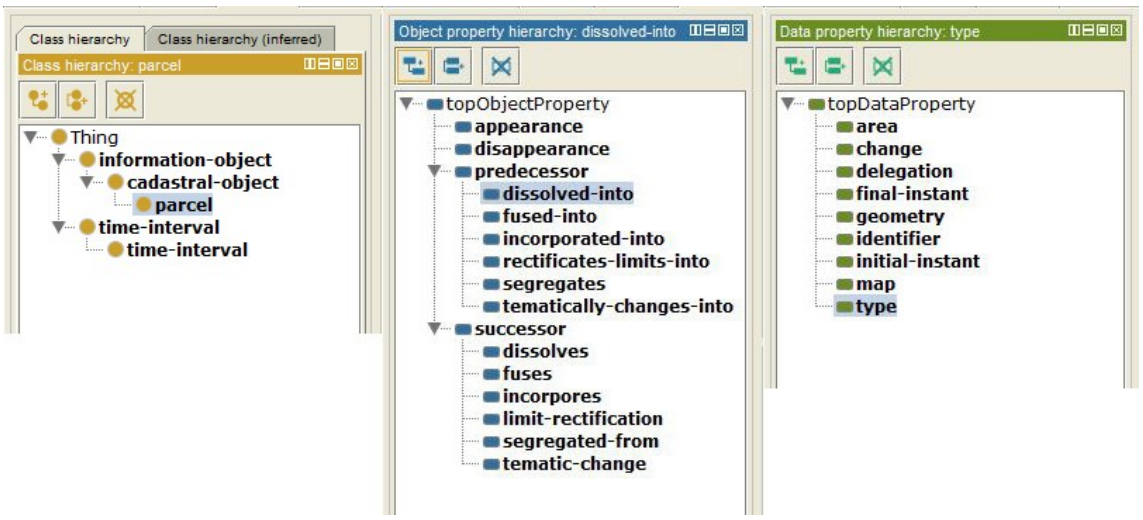

Figura 11. Árbol con las definiciones del vocabulario

- *type*¸ tipo de parcela: rural, urbana…

Por motivos de sencillez operacional se eliminaron de JDO todas las clases, propiedades y características que no iban a ser utilizados en HCO. En caso de buscar la interoperabilidad completa o querer ampliar el vocabulario sólo habría que volver a incluir los elementos eliminados.

Como paso final para la creación del vocabulario se utilizó el razonador ontológico de Protégé para detectar posibles faltas de coherencia dentro del trabajo realizado.

Finalmente, y al igual que hicieron Lacasta et al. (2014), se colocó la ontología en un *hub* de datos para que sea accesible y pueda ser modificado por cualquier usuario para sus propios fines. El *hub* elegido es "github" debido a que con una simple URI ofrece los datos en "crudo" para que las aplicaciones de SPARQL puedan leerlo sin problemas. HCO se puede encontrar en el siguiente enlace:

https://raw.github.com/AlvBch/HCO/MasterThesis/HCO.owl

4.2.2. Prueba de consistencia

Para comprobar que el vocabulario está bien construido y que la series de tripletas que ser formarán son consistentes se ha realizado un pequeño ejemplo, con cinco parcelas al azar.

Estas parcelas han sido introducidas manualmente en Protégé con todos sus datos, características y relaciones (Figura 13).

Cabe destacar en este punto que Protégé no permite la introducción automática de

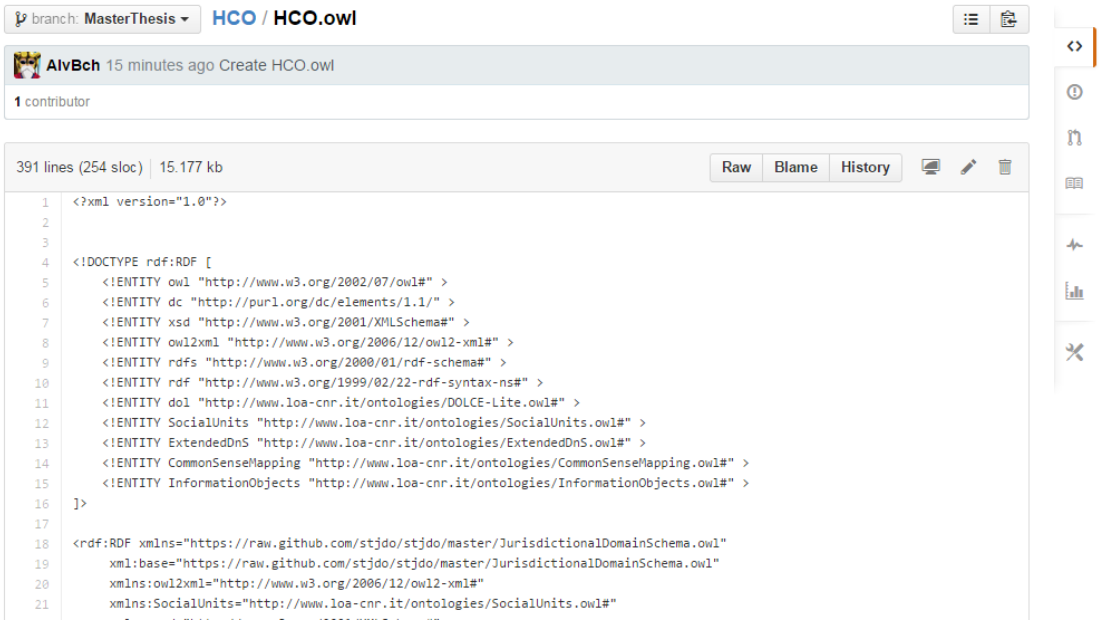

Figura 12. *Hub* de datos que alberga el vocabulario HCO

los datos, por ello se tuvo que buscar otra herramienta para la unión del SGBDR con el vocabulario.

Antes de comenzar a hacer consultas se fijaron los prefijos que se usarán en las mismas. Un prefijo de una consulta SPARQL no es más que el designio de direcciones de ontologías por medio de unas pocas letras. Estos prefijos facilitan mucho la tarea de consulta, además de agilizarla. A continuación se muestran los prefijos utilizados en las consultas al ejemplo:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX hco: <https://raw.github.com/AlvBch/HCO/MasterThesis/HCO.owl#>
PREFIX dns: <https://www.loa-cnr.it/ontologies/ExtendedDnS.owl#>
PREFIX dolce: <https://www.loa-cnr.it/ontologies/DOLCE-Lite.owl#>
PREFIX geo: <http://www.w3.org/2003/01/geo/wgs84_pos#>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX locn: <http://www.w3.org/ns/locn#>
```

Cabe destacar en este punto que Virtuoso se reserva una serie de prefijos para su uso propio. Mediante el prefijo bif (*Built-In functions*) Virtuoso ofrece una serie de funciones de gran interés al usuario, entre las que se encuentra el GeoSPARQL:

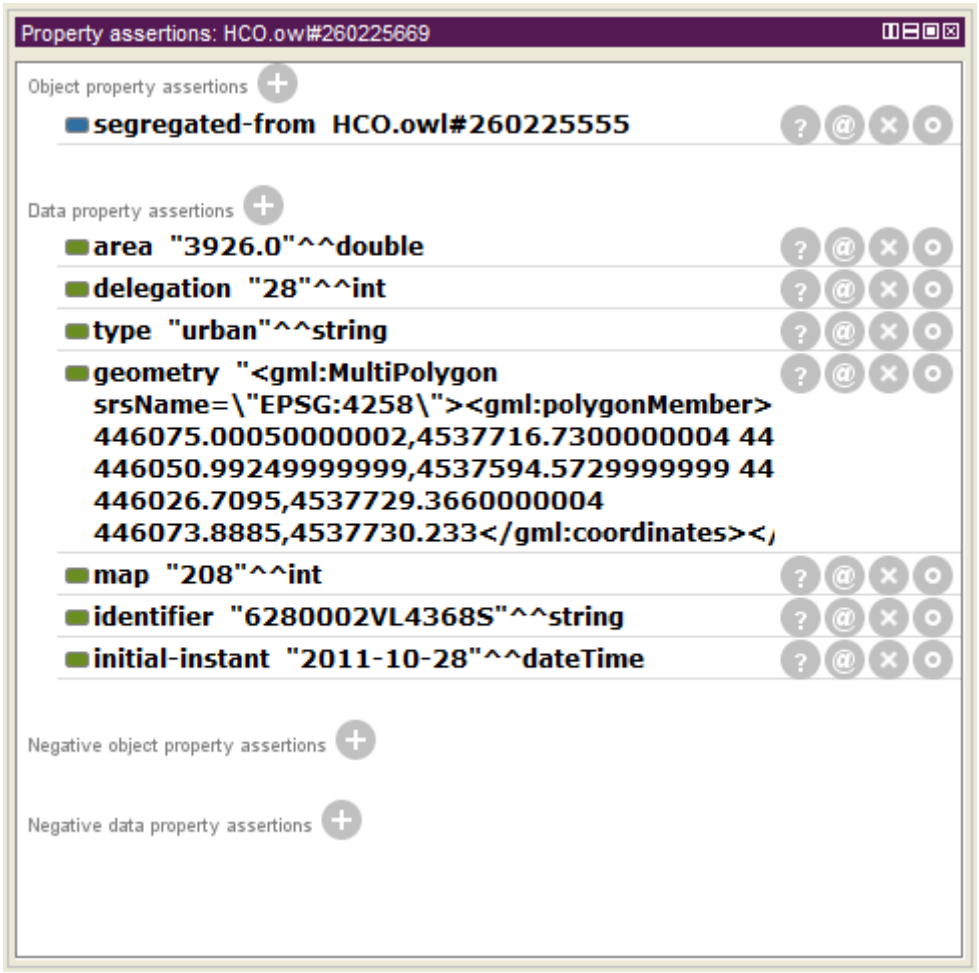

Figura 13. Parcela de ejemplo en Protégé

http://docs.openlinksw.com/virtuoso/functions.html

Las consultas realizadas, una vez fijados los prefijos, comenzaron buscando mostrar información básica (clases y subclases):

```
SELECT ?subject ?object
   WHERE { ?subject rdfs:subClassOf ?object }
```

Datos más complejos sobre las parcelas introducidas:

```
SELECT ?parcela ?inicio ?fin
      WHERE { ?parcela hco:initial-instant ?inicio .
              ?parcela hco:final-instant ?fin }
```

Datos con restricciones temporales:

```
SELECT ?parcela ?inicio ?fin
      WHERE { ?parcela hco:initial-instant ?inicio .
              FILTER ( ?inicio > "2011-01-01" ^^ xsd:dateTime )
          ?parcela hco:final-instant ?fin . }
```

Al concluir esta pequeña serie de pruebas y consultas a la base de datos semántica de prueba se pudo asegurar que el vocabulario era consistente y que funcionaba correctamente, por lo que se pudo pasar a la siguiente fase.

4.3. Adaptar datos

Llegado este punto del trabajo de fin de master se hizo necesaria la unión de la base de datos relacional sobre el linaje catastral con el vocabulario creado para obtener una base de datos semántica.

Los estudios realizados dieron como resultado que no existe un programa que fuera capaz de guardar el SGBDR en RDF aplicando el vocabulario descrito (también en RDF). Por tanto, era necesario crear un programa explícitamente desarrollado para este estudio.

Gracias al ejemplo creado en Protégé se podía estudiar y conocer la estructura que debía presentar la base de datos semántica, el objetivo final del programa a crear.

Como paso previo a la creación del programa se modificó la base de datos de origen:

- Fusionar ambas tablas de datos en una sola: gracias a esta operación el programa a realizar sólo tiene que leer un dato de origen lo cual le otorga sencillez y operatividad

- Reestructurar el orden de datos: con el nuevo orden, los datos coinciden con el orden esperado en la ontología, lo cual también facilita el desarrollo del programa

Lo que en principio parecía una fase rápida terminó convirtiéndose en uno de los puntos que más trabajo supuso, requiriendo 66 consultas SQL diferentes, alcanzando algunas grados de complejidad bastante altos.

4.3.1. Insertar geometría

Al cargar los datos espaciales en pgAdmin se pudo comprobar que los datos no tenían cargado el sistema de referencia deseado: EPSG 4326 (geográficas WGS84).

Este problema no tenía influencia cuando se pretendía el estudio de los datos, pero ahora estos datos se convertirán en el producto final, por lo que es necesario que tengan el sistema de referencia.

El proceso de inserción de geometría se realizó mediante GeoKettle, herramienta ETL, que cargó los datos al servidor postgreSQL de pgAdmin una vez finalizado el proceso.

El flujo de trabajo que se creó constaba de tres pasos:

- Cargar el shapefile
- Establecer el sistema de coordenadas
- Exportar la tabla a postgreSQL

Cada vez que se establecían los datos para el paso anterior se podía establecer el flujo (llamados *hops*) que une cada paso.

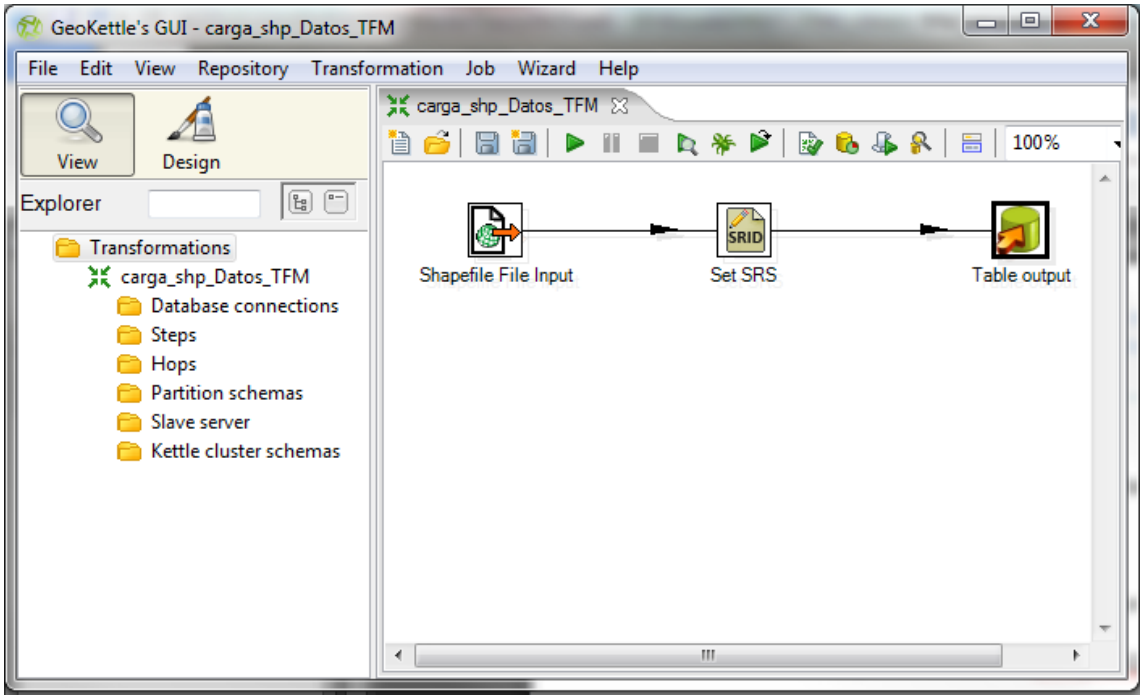

Figura 14. Flujo de trabajo en GeoKettle

El primer paso (carga del shapefile) sólo requería que se estableciera la ruta en la que se encuentra el archivo shapefile.

Para establecer el sistema de coordenadas primero se seleccionó el campo en el que iría la geometría (era necesario que el campo fuera geométrico) y después el sistema de coordenadas a cargar en código EPSG. Dicho código se extrajo del archivo .prj que acompaña al shapefile de las parcelas: ETRS89 UTM 30N, cuyo código EPSG es 25830.

El último paso establecía los datos de conexión necesarios para exportarlos al servidor postgreSQL. En la tabla 3 se pueden apreciar los datos de conexión requeridos por GeoKettle.

Tras este paso GeoKettle creó y mostró un diálogo en el cual se pudo observar los campos que se iban a crear en la tabla de destino. Mediante el botón "SQL" (consulta SQL que ejecutaría el proceso solicitado) se pudo editar dichos campos para que su orden fuera el deseado y sus tipos de datos los correctos (fue necesario editar el orden y los tipos de datos de 5 campos).

Cuando todos los datos estuvieron correctos se ejecutó la transformación, agregando el sistema de coordenadas y la tabla a la base de datos postgreSQL.

4.3.2. Fusionar bases de datos

Como ya se ha dicho, la fusión de las tablas de origen de datos es un paso muy importante para el estudio ya que si no se realiza correctamente los datos estarán corruptos y las consultas SPARQL arrojarán datos erróneos. Por ello se puso especial

Tabla 3. Datos de conexión a la base de datos desde GeoKettle

Nombre de la Conexión (*Connection Name*)	Datos_TFM
Tipo de Conexión (*Connection Type*)	PostgreSQL
Acceso (*Access*)	Native (JDBC)
Nombre del Hospedador (*Host Name*)	localhost
Nombre de la Base de Datos (*Database Name*)	postgres
Número del puerto (*Port Number*)	5432
Nombre del Usuario (*User Name*)	Alv
Contraseña (*Password*)	AlvTFM2015

cuidado en este proceso y se realizaron repetidas comprobaciones de que los datos se transformaron correctamente.

Antes de comenzar con este paso se ha de tener en cuenta que para fusionar las dos tablas de datos hace falta que todos los elementos de ambas tablas estén relacionados unívocamente, es decir, no existan ambigüedades que puedan hacer que se asignen características que no le correspondan a los datos.

Podría parecer que la respuesta evidente para esta cuestión es la referencia catastral, pero se comprobó que en la mayoría de los casos la referencia catastral del padre se repetía en uno de los hijos. Por tanto, para la asignación de las correspondencias entre los datos de la tabla parcelas y la tabla linaje se ha utilizado el identificador interno ("ninterno") que asigna el proceso descrito por Moreno et al. (2014). Mediante la comprobación de la referencia catastral, las fechas de validez y el área de la parcela, se comprobó que a cada parcela se le asignaba el número interno correspondiente en ambas tablas.

El desarrollo de esta fase fue bastante complejo (como ya se ha dicho, se realizaron 66 consultas SQL), por lo que a continuación se enuncian los pasos dados para llegar a conseguir la fusión de la base de datos, con una pequeña explicación de los mismos cuando sea necesario. A modo de ejemplo algunos pasos vendrán acompañados de su consulta SQL para mostrar el grado de complejidad alcanzado por las mismas.

4.3.2.1. Adaptar base "parcelas"

- Realizar una copia exacta (con los índices y restricciones creados) de los datos de origen para manipular la copia de los datos

```
CREATE TABLE _parcelas
    (LIKE parcelas INCLUDING defaults INCLUDING constraints INCLUDING
      indexes);
```

- Copiar todos los datos de las tablas anteriores a las nuevas
- Crear extensión postGIS, creando así una base de datos espacial, esta consulta crea una tabla nueva en el servidor que contiene la información del sistema de referencia espacial
- Crear índice espacial
- Añadir columna que almacenará las fechas en formato "*date*"
- Almacenar las fechas en formato "*date*"

```
UPDATE _parcelas SET fecha_baja = to_date(fechabaja, 'YYYYMMDD')
      WHERE fechabaja <> '99999999';
```

- Eliminar las columnas con el tipo de dato erróneo para la fecha
- Se comprueba si el dato que unirá las bases de datos es único y está correcto: consulta para buscar registros repetidos en ninterno
- Crear el dato que almacenará las geometrías en formato de texto
- Cargar la geometría en texto

```
UPDATE _parcelas SET geometry = ST_AsText(the_geom);
```

- Eliminar las columnas que no serán utililizadas por el programa para el paso a RDF

4.3.2.2. Adaptar base "linaje"

- Realizar una copia exacta de los datos de origen para manipular la copia de los datos
- Copiar todos los datos de las tablas anteriores a las nuevas
- Añadir columna que almacenará las fechas en formato "*date*"
- Almacenar las fechas en formato "*date*"
- Eliminar columna ninterno, puesto que se matizará en "ninterno_padre" y "ninterno_hijo"
- Crear columnas "ninterno_padre" y "ninterno_hijo"

```
ALTER TABLE _linaje
   ADD COLUMN ninterno_padre bigint, ADD COLUMN ninterno_hijo bigint;
```

- Asignar el ninterno de parcela a ninterno_padre y/o ninterno_hijo sí y sólo sí las referencias catastrales, su áreas y sus fechas coinciden

```
UPDATE _linaje l SET ninterno_hijo =
   ( SELECT p.ninterno FROM _parcelas p
     WHERE p.refcat = l.hijo AND p.fecha_alta = l.fechaini_hijo AND
     p.area = l.area_hijo LIMIT 1);
```

- Comprobación si todos los números de ninterno se han guardado de manera correcta, pues deberían de ser diferentes,

```
SELECT COUNT (DISTINCT(ninterno_padre)) AS dpadre,
      COUNT (DISTINCT(ninterno_hijo)) AS dhijo FROM _linaje;

SELECT COUNT (ninterno_padre) AS dpadre,
      COUNT (ninterno_hijo) AS dhijo FROM _linaje;
```

si los números coinciden se habrá hecho de manera correcta

- También se comprueban que no existan datos en blanco cuando debería haber dato con una consulta condicional

```
SELECT
   CASE WHEN padre is not null and ninterno_padre is null THEN 'PADRE'
        WHEN hijo is not null and ninterno_hijo is null THEN 'HIJO'
        ELSE 'OK'
   END
   FROM _linaje;
```

- Comprobar que en el caso de que padre e hijo tengan la misma referencia catastral carguen su ninterno correspondiente (que es diferente)

- Buscar si algún hijo ha cogido los datos del padre por error, relacionando los datos obtenidos con la fecha

- Comprobar si los ninternos corresponden con el su correspondiente referencia catastral

- Mediantes la anidación de CASE se buscan datos incorrectos

- Eliminar las columnas que no serán utilizadas por el programa para el paso a RDF

4.3.2.3. Crear tabla fusión

- Se crea una nueva tabla donde se unirán los datos de originales para la salida en CSV, el orden y tipo de los datos se ha ajustado a las necesidades futuras

- Se crean sus índices y restricciones

- Se agregan los datos a la tabla procedentes de la tabla

- Al agregar los datos de tipo se “traducen” los códigos tal como se espera que figuren en la base de datos semántica

```
INSERT INTO _data

      SELECT p.ninterno, p.refcat, p.mapa, p.delegacio, p.municipio,
            CASE  WHEN p.tipo = 'R' THEN 'rural'
                  WHEN p.tipo = 'U' THEN 'urban'
                  ELSE 'X'
            END AS tipo, p.area, p.geometry, p.fecha_alta, p.fecha_baja
      FROM _parcelas p ;
```

- Se introducen los datos de cambio, eliminando los códigos numéricos y sustituyendo por su valor textual correspondiente, obtenidos de la tabla linaje

- El cambio se introduce tanto como cambio en relación con el predecesor como cambio en relación con el sucesor (que será el valor inverso, por ejemplo: segregates/segregated-from)

- Se comprueba que los datos de cambio están correctamente grabados

- Se detecta y elimina un dato de cambio predecesor que posee el código inverso al que le corresponde

- Se aúnan todos los ninterno_hijo de cada padre en un *array* de una fila, separados por espacios grabándose todos ellos en una única celda del individuo, en caso de que existan hijos

```
UPDATE _data d SET predecesor = (
      SELECT
            CASE  WHEN d.cambio_suc IS NULL THEN NULL
                  WHEN d.cambio_suc = 'appearance' THEN NULL
                  ELSE array_to_string (
                        ARRAY (SELECT l.ninterno_padre FROM _linaje l
                              WHERE l.ninterno_hijo = d.ninterno ),
                              ' ', NULL )
            END
      LIMIT 1 );
```

- Se comprueba que el número de hijos guardado sea el mismo que el número de hijos que realmente posee el padre

```
SELECT COUNT (l.ninterno_hijo),
   ARRAY_TO_STRING (
               ARRAY (SELECT l.ninterno_hijo FROM _linaje l WHERE
               l.ninterno_padre = d.ninterno ),
               ' ') AS nhijo, d.ninterno
      FROM _linaje l, _data d WHERE l.ninterno_padre = d.ninterno
      GROUP BY d.ninterno;
```

- Consulta que devuelve todos los datos de la table

```
SELECT * FROM _data ORDER BY cambio_pre;
```

- Exportar datos a CSV para ser usados en el programa creado específicamente para aplicar la ontología a los datos. Las columnas están separadas por ";" y el texto citado está entre comillas dobles ' " ', la codificación seleccionada fue *Unicode UTF-8*, se añaden las cabeceras de las columnas

El archivo CSV exportado contaba con **11246 filas** (incluyendo la cabecera) y **14 columnas**, con un total de **107.318 datos** registrados.

Durante la creación de la tabla de datos que aunaba los datos de origen se descubrió un fallo en uno de los datos: el cambio de un predecesor tenía un valor que no le correspondía (correspondía a un código que era un cambio de sucesor); se

comprobó que el dato venía erróneo desde la tabla linaje y no de la introducción de los datos.

Cuando se manipulan bases de datos de gran tamaño de la manera que se hizo, siempre hay que tener en cuenta que en caso de error es más funcional borrar y empezar de nuevo a continuar intentando subsanar el error, ya que existen muchos procesos internos del programa que son desconocidos para el usuario.

4.4. Aplicar ontología

Con todos los elementos preparados por separado, se procedió a continuación a crear la herramienta que sirve para su unión.

Existe un gran número de programas que permiten el paso de cualquier tipo de archivo de datos a RDF y prácticamente todos ellos son capaces de manipular CSV. El problema encontrado es que, a pesar de realizar la transformación, no permite hacerla de la manera que es necesaria para el estudio: aplicando una ontología y guardando una base de datos relacional de manera semántica. Es debido a esto por lo que se tomó la decisión de diseñar un programa específico para este proyecto y esta tarea.

Para ello fue necesario el estudio del lenguaje de programación Java y la familiarización con la aplicación Eclipse, con la que se realizó el programa.

Durante la creación de dicho programa siempre se tuvo presente la ontología que se quería aplicar y el diseño de los datos de entrada. Es por ello que no sería de utilidad para ninguna otra ontología ni ningún otro diseño de datos de entrada. En el Anexo I se encuentra detallado el modelo de datos de entrada, indicando el orden que deben de tener los campos. El tipo de dato de los mismos es indiferente, ya que al exportar los datos a CSV se convierten en texto.

El resultado final de esta fase, es un programa que carga un fichero CSV con los datos de linaje catastral y un fichero RDF con la cabecera de la ontología para escribir una base de datos semántica. Este programa se creó bajo los principios de la licencia *Creative Commons Attribution-NonComercial-ShareAlike* (CC BY-NC-SA 3.0 https://creativecommons.org/licenses/by-nc-sa/3.0/deed.es_ES) y está disponible para cualquier usuario que lo necesite.

4.4.1. Programa CSV a RDF

El programa que se creó consta de cuatro clases y se apoya en una librería externa que se encarga de gestionar los ficheros CSV: javacsv.jar.

La biblioteca referenciada se encarga de las tareas de lectura y escritura de CSV, aunque sólo se usó el código de lectura.

Las clases creadas fueron:

- **main**, llama a las funciones principales de la aplicación, carga y exporta los datos
- **Parcela**, contiene los campos de las características de las parcelas
- **csvLeeEscribe**, alberga las funciones de lectura y escritura de CSV
- **txtLeeEscribe**, escribe el texto concreto de la base de datos semántica cargando los datos guardados en "Parcela" (también tiene funciones de lectura de texto)

En primer lugar se buscó por la red aplicaciones similares a lo que se pretendía hacer que estuvieran disponibles de manera abierta.

A este respecto se encontró el trabajo de Martínez, F. J., 2006, que bajo una licencia *Creative Commons* de reconocimiento y no comercial compartió un código de lectura de CSV. También se encontraron los trabajos de Villalobos, J., compartidos sin licencia, que muestran cómo funcionan las herramientas de lectura y escritura de las librerías nativas de Java.

Para que el programa pueda ser desarrollado correctamente y evitar problemas según se vaya depurando, se comenzó el mismo con la creación de la clase que definirá los elementos que se van a utilizar: Parcela.

En esta clase se definieron 14 elementos privados de texto (tipo *string*), con el orden de la base de datos, en los que se cargar los datos leídos de la tabla en CSV. También se crearon los métodos para establecer y llamar a los datos.

A continuación se pueden observar los códigos de creación de los datos y un ejemplo de los métodos de establecimiento y devolución de un dato.

La siguiente clase creada, basada en el trabajo de Martínez (2006), fue csvLeeEscribe, con el apoyo de la biblioteca externa de lectura y escritura de CSV (junto con las librerías nativas de Java en Eclipse).

```
/**
 * Datos de la tabla  */
private String ninterno;
private String refcat;
private String mapa;
private String delegacion;
private String municipio;
private String tipo;
private String area;
private String geometry;
private String fechalta;
```

```
private String fechabaja;
private String cambiopre;
private String cambiosuc;
private String sucesor;
private String predecesor;

/**
 * Métodos que devuelven y establecen los datos
 */
public String getNinterno() {
   return ninterno;
}
public void setNinterno(String nint) {
   this.ninterno = nint;
}
```

Esta clase posee dos métodos cuyas funciones están acotadas con *try/catch* para contener posibles problemas que surjan en la aplicación y escribir dicho error mediante la consola del sistema:

- Una lista (ArrayList) de lectura de CSV:
    - datos de entrada
        - ruta del fichero a leer (String)
        - delimitador de datos en el fichero (String)
    - localiza el fichero introducido - FileReader(fichero) -
    - crea el lector de CSV introduciendo fichero y delimitador – CsvReader(fichero, delimitador) -
    - lee la cabecera del archivo si la tuviera
    - gracias a un bucle while:
        - lee los datos por filas
        - almacena cada línea del fichero leída en un elemento de la clase “Parcela” – Parcela.setDato(datoLeido) -
        - guarda la Parcela en el ArrayList – lista.add(Parcela) -
    - devuelve la lista - return ArrayList –

A continuación se muestran unas partes del código que muestran el funcionamiento del método.

```
//Leer los regristros
System.out.println("___DATOS___");
```

```
    while(lee.readRecord()) {
          //Se puede usar get con el nombre de la cabecera o por
          posición
          String ninterno = lee.get(head[0]);
          String refcat = lee.get(head[1]);
          String mapa = lee.get(head[2]);
          String delegacion = lee.get(head[3]);
[…]
    //Almacenar el objeto
    Parcela parc = new Parcela();
    parc.setNinterno(ninterno);
    parc.setRefcat(refcat);
[…]
                        if (lee!=null){
                              lee.close();
                        }
                  } catch(IOException e) {
                        System.out.println("--ERROR--");
                        System.err.println(e);
                  }
          return listaReg;
    }
```

El siguiente método es:

- Un void de escritura en CSV:
    - datos de entrada
        - List con los datos
        - ruta donde escribir el fichero de salida (String)
        - delimitador de los datos (String)
    - crea el escritor de archivos - FileWriter(ruta) -
    - crea el escritor de CSV introduciendo el fichero donde escribir y el delimitador - CsvWriter(fichero, delimitador) -
    - escribe la línea de la cabecera
    - mediante un bucle for escribe todas las líneas de la lista de entrada en el fichero - CsvWriter.write(dato) -

La clase txtLeeEscribe está compuesta por dos métodos, necesarios para la lectura y escritura de archivos de texto.

Al igual que los métodos descritos anteriormente, trabajan con *try/catch* para contener y evaluar, en medida de lo posible, los posibles fallos que pueda dar la aplicación.

- Un *string* de lectura que carga los datos a leer en un elemento de texto:
    - datos de entrada
        - elemento de tipo S*tring* que contiene la ruta del archivo a leer
    - se crean los *String* que leerán cada línea y almacenarán lo leído
    - crea el lector de ficheros - *FileReader(fichero)* –
    - y crea el lector que irá guardando lo leído en éste
        - *BufferedReader(FileReader(fichero))* -
    - con un bucle *while*:
        - lee una línea - *readLine ()* -
        - si se había leído una línea con anterioridad (es decir el *String* auxiliar no es nulo) se añade esta línea elemento auxiliar junto con un retorno de carro "\n"
    - devuelve el *String* guardado con las líneas leídas - *return String* –

Se estimó más funcional para el programa que la cabecera de la base de datos con los prefijos y los entornos de la ontología a escribir se añadiera por separado en un fichero. El programa leería dicho fichero y agregaría los datos a continuación con la ontología ya agregada. Esta medida facilitó en gran medida el código e hizo más sencillos los procesos del mismo.

- Método tipo *void* para la escritura del archivo de texto:
    - datos de entrada
        - *String* con la ruta donde guardar el fichero
        - *String* con la cabecera que se desea en el archivo
        - *List* con los datos del CSV
    - crea el escritor de archivos - *FileWriter (ruta)* -
    - el escritor que guarda las líneas de éste
        - *BufferedWriter(FileWriter(ruta))* -
    - y por último el elemento que escribe lo guardado en el anterior
        - *PrintWriter(BufferedWriter(FileWriter(ruta)))*
    - primero escribe la cabecera introducida

- mediante un bucle *for* escriben todos los datos en la manera que deben figurar en la base de datos semántica
  - comando *switch* para desambiguar las dos aproximaciones para cargar las relaciones de linaje entre padres e hijos
    - *case* de desaparición o aparición
    - *default*: mediante un *split* se cargan todos los ninternos que tenían relación con el dato en un *Array* (*String[]*) y uno a uno se escriben con el cambio realizado
- escribe las líneas finales del fichero que cierran la ontología
- y cierra los escritores

Con todas las clases creadas y los respectivos métodos listos, llegó el momento de crear la clase principal main con un método que llamase a los métodos y comprobar si funcionaban correctamente.

- método *void*
  - datos de entrada
    - lista *String[]* con los datos de lectura y escritura necesarios
      - ruta de lectura del CSV
      - delimitador del CSV
      - ruta de lectura de la cabecera (en .txt o .rdf)
      - ruta en la que escribir la base de datos semántica
  - asigna los valores de entrada a *String*
  - sucesión de *try/catch* que llama a los métodos previamente creados
    - csvLector
    - txtLector

txtEscritor Las líneas de código que muestran el funcionamiento principal del método son:

```
public void txtEscritor(String pathFileTxt_E, String PrevText, List
csvData)
   throws Exception{
   //Escritor de .txt
```

```
   FileWriter txtWriter = null;
   BufferedWriter txtBwriter = null;
   PrintWriter txtPwriter = null;

   Parcela parcel = null;
[…]
try{
   File txtFich_E = new File (pathFileTxt_E);

   txtWriter = new FileWriter(pathFileTxt_E);
   txtBwriter = new BufferedWriter (txtWriter);
   txtPwriter = new PrintWriter (txtBwriter);

   txtPwriter.write(PrevText);

   for(int i=0;i<csvData.size();i++){
      parcel = (Parcela)csvData.get(i);
      txtPwriter.append("\n\n\n\n\t<!—
            https://raw.github.com/AlvBch/HCO/MasterThesis/HCO.owl#"
             + parcel.getNinterno() + " -->\n\n");
      txtPwriter.append("\t<NamedIndividual rdf:about=\"&HCO;"
             + parcel.getNinterno() + "\">\n");
[…]
switch(parcel.getCambiopre()){
      case "disappearance":
            txtPwriter.append("\t\t<HCO:" + parcel.getCambiopre()
                  + "/>\n");

      default:
            if(parcel.getSucesor() != ""){
                  String[] suc = parcel.getSucesor().split("\u0020");
                  for(int j=0;j<suc.length;j++){
                        txtPwriter.append("\t\t<HCO:"
                              + parcel.getCambiopre()
                              + " rdf:resource=\"&HCO;" + suc[j]
                              + "\"/>\n");
                  }
            }
      break;
[…]
      if (txtPwriter!=null && txtBwriter!=null){
            txtPwriter.close();
            txtBwriter.close();
      }
}catch (IOException e){
      System.out.println("--ERROR--");
      System.err.println(e);
}
```

Con esta clase se terminó la aplicación y se comenzó con la transformación de datos de CSV a RDF.

Se exportó un pequeño ejemplo de aproximadamente una docena de elementos de la base de datos relacional para realizar pruebas y comprobar que la aplicación funcionaba antes de intentar hacerlo funcionar con la base de datos al completo.

Hicieron falta 4 versiones de la aplicación para conseguir una depurada y estable. El proceso de depuración se realizó, en su mayoría, gracias a las herramientas de "*debug*", "parada" y de "ejecución paso por paso" que presenta Eclipse.

4.4.2. Base de datos ontológica

Todos los pasos previos se realizaron con el objetivo de poder crear una base de datos semántica en la que se le aplica una ontología a una base de datos relacional. En esta fase se siguieron los procesos necesarios para la obtención del objetivo principal de este estudio: una base de datos semántica con información del linaje catastral.

4.4.2.1. Creación

Finalmente, con todos los archivos de origen exportados se pudo ejecutar la aplicación desde la herramienta *debug* de Eclipse con la configuración que se observa en la figura 15.

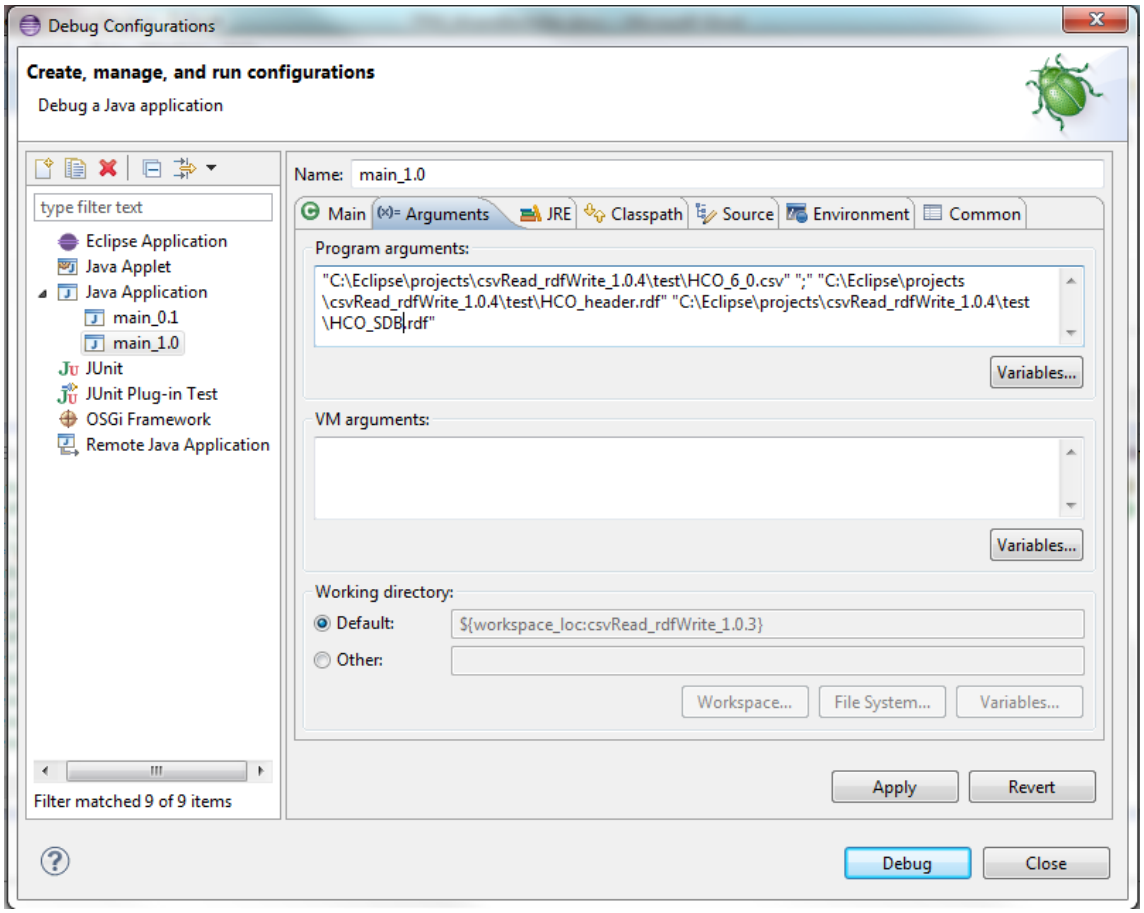

Figura 15. *Debug Configuration* para guardar la base de datos semántica

Tabla 4. Datos para crear la base de datos semántica

Datos de entrada	HCO_RDB.csv
Delimitador de datos	;
Cabecera	HCO_header.rdf
Archivo exportado	HCO_SDB.rdf

La tabla 4 muestra los datos que se introducen en la aplicación para su procesado

Este proceso se ejecutó en 6,32 segundos guardando 144.222 tripletas de datos.

Los datos fueron guardados en RDF porque así se especificó en la ruta asignada de guardado. La aplicación lo único que hace es guardar un texto con el nombre que se le indique, es el usuario el que especifica el tipo de fichero .rdf.

Cabe destacar que existen diferencias entre la estructura de una base de datos semántica escrita en RDF y la de la misma base de datos escrita en otro lenguaje (OWL, por ejemplo). La aplicación fue desarrollada con la estructura y el formato del RDF, por lo que si se especificase una extensión diferente en el archivo, este no estaría guardado en la nueva extensión, sino que, seguramente, arrojaría errores al intentar trabajar con el mismo.

4.4.2.2. Carga a Virtuoso

La base de datos (ahora semántica) se cargó en el servidor de virtuoso mediante los siguientes:

- Instalación de Virtuoso
    - descomprimir los archivos binarios en la ruta deseada
    - mediante la configuración avanzada de las propiedades del sistema, se agrega una variable de entorno
    - se establece la variable *%VIRTUOSO_HOME%* a la ruta donde se descargaron los archivos
    - se edita la variable "*Path*" para agregar *%VIRTUOSO_HOME%/bin; %VIRTUOSO_HOME%/lib;*
- Abrir Virtuoso
    - abrir la consola de ms-DOS ejecutando la consola "cmd" como administrador
    - se introduce el comando *cd %VIRTUOSO_HOME%/database*, o la variable de entorno que se definió al instalar el programa

- existen dos maneras de ejecutar el programa
  - creando un servicio de Windows mediante el comando: *virtuoso-t+service create+instance "Virtuoso"+configfile virtuoso.ini*
  - o bien navegando hasta la carpeta *database* donde se encuentra el archivo .ini e introduciendo el comando *virtuoso-t -f* (es necesario poner un espacio entre "t" y "-f") (la extensión -f hace que arranque el programa)
- la herramienta de administración de Virtuoso se puede cargar en un navegador a través del link: http://localhost:8890/conductor/
- se introduce usuario y contraseña
  - usuario: *dba*
  - contraseña: *dba*
- ya se puede comenzar a trabajar con OpenLinkVirtuoso

- Cargar datos al servidor
  - entrar en la pestaña *Linked Data*
    - "*Schemas*" para introducir el IRI (Identificador de Recurso Internacionalizado en sus siglas en inglés) del vocabulario creado y almacenado en el *hub* de datos
    - "*Quad Store Upload*" para cargar los datos al servidor
      - pulsar el botón "*Choose File*" para introducir un archivo local
      - seleccionar "*Resource*" para introducir un URI
  - para cargar la base de datos al localhost se pulsa en "*Upload*"
- Eliminar datos del servidor para subir una nueva versión de la base de datos semántica
  - dentro de la pestaña *Linked Data*
    - se entra en la pestaña *"Graphs*"
      - dentro de la subpestaña "*Graphs*" se puede encontrar una tabla con todos los *Graph* que tiene

introducidos Virtuoso, el Graph cargado localmente tiene un nombre por defecto de: http://localhost:8890/DAV (en caso no haberse cargado con un nombre por defecto se debe buscar el nombre cargado); pulsando la opción "*Delete*" se eliminará la base cargada permitiendo así cargar una nueva en el servidor

Para terminar el programa simplemente hay que pulsar "control+c" dentro de la ventana de la consola

4.5. Consulta

Con la base de datos semántica preparada en OpenLinkVirtuoso, se pasó a la fase final del estudio: la realización de consultas para comprobar su funcionalidad.

El objetivo de esta fase era doble: por un lado se pretendía comprobar las ventajas que aporta la web semántica a una base de datos relacional y por otro lado se quería verificar que los procesos anteriores de preparación y carga de la ontología se realizaron correctamente.

Las consultas se realizaron a la base de datos relacional de linaje catastral original (los datos tal cual fueron recibidos) mediante el programa pgAdmin y en lenguaje SQL y a la base de datos semántica, de reciente creación, mediante la herramienta web de OpenLinkVirtuoso, en SPARQL.

Las consultas en pgAdmin se realizan mediante su herramienta de consulta, accesible desde la vista general del programa.

Para realizar consultas SPARQL en Virtuoso es necesario navegar hasta *Linked Data* y dentro de la pestaña a *SPARQL.*

A continuación se enumeran algunas de las consultas realizadas, con su correspondiente consulta en el otro lenguaje más una pequeña explicación de lo que se busca con la misma.

Para realizar las consultas en SPARQL se debe tener en cuenta que se cargaron los mismos prefijos que se indicaron en el epígrafe 4.2.2 "Prueba de consistencia".

Tabla 5. Comparación de consultas SQL y SPARQL

SQL	SPARQL	Descripción
-	`SELECT ?class ?superclass` `WHERE { ?class rdfs:subClassOf ?superclass }`	Seleccionar todas las clases y subclases. Análisis de la base de datos semántica
`SELECT COUNT(ninterno) FROM _data` `WHERE geometry IS NOT NULL;`	`SELECT (COUNT(DISTINCT (?s)) AS ?no)` `{ ?s geo:geometry ?o }`	Cuenta el número de datos que tienen geometría: 11.246
`SELECT ninterno FROM _data;`	`SELECT ?data` `WHERE { ?data rdf:type owl:NamedIndividual }`	Obtener un dato de las parcelas, por ejemplo el número interno

`SELECT ninterno FROM _data`
`WHERE cambio_suc='segregated-from'`
`AND area > 30000;`
	`SELECT ?parcel ?area`
	`WHERE { ?parcel hco:segregated-from ?x .`
	`?parcel hco:area ?area .`
	`FILTER ( ?area > "30000"^^xsd:double ) }`
		Seleccionar parcelas segregadas con un área mayor que 3 ha

`SELECT ninterno FROM _data`
`WHERE fecha_baja < '2012-01-01';`
	`SELECT DISTINCT ?parcel ?date`
	`WHERE { ?parcel rdf:type owl:NamedIndividual.`
	`?parcel hco:final-instant ?date`
	`FILTER (?date < "2012-01-01"^^xsd:dateTime) }`
		Parcelas que desaparecen antes de una fecha determinada

Cuando se procedió a realizar consultas espaciales surgió un problema con la versión utilizada de Virtuoso (la 7.10.3207): esta versión sólo soportaba consultas espaciales en las cuales se manipulase un punto y una superficie, no dos superficies; por lo que no era posible definir una *bounding box* y consultar qué parcelas se encuentran en la misma, función que se pretendía implementar en un trabajo futuro de un visor en línea de los datos de linaje catastral.

Para comprobar que la componente espacial de la base de datos semántica funciona correctamente y los datos están bien guardados se barajó la posibilidad de calcular el centróide de las parcelas catastrales y comprobar si dicho centróide se encontraba dentro de los límites definidos. Sin embargo, no existe una función en esta versión que realice el cálculo del centróide.

Por tanto el cálculo del centróide se realizó en pgAdmin mediante consulta SQL y se exportó una nueva base de datos en CSV. Esta tabla se transformó a RDF mediante la herramienta específicamente creada para este trabajo de fin de máster. En la nueva base de datos semántica la “geometría” es nada más que el centróide de las parcelas.

Para llevar a cabo esta tarea se creó una nueva columna en la tabla “_parcelas” que contendría a los centóides llamada “geo_centroid".

Mediante la consulta:

```
UPDATE _parcelas SET geo_centroid = ST_Centroid(geometry);
```

se calcularon y guardaron los centróides de las parcelas en la tabla.

Finalmente se actualizó la tabla donde se fusionaban los datos con los nuevos datos para “geometría”.

```
UPDATE _data3 d SET geometry = st_astext(
      (SELECT geo_centroid FROM _parcelas p
            WHERE p.ninterno = d.ninterno));
```

Una vez cargados los datos en Virtuoso se pudo continuar realizando consultas espaciales teniendo en cuenta sus limitaciones.

Una combinación de los filtros anteriores da como resultado unos datos de gran interés cuando se trata el linaje catastral:

Tabla 6. Consultas espaciales y multifiltro en SQL y SPARQL

SQL	SPARQL	Descripción

`SELECT ninterno, refcat, area`
`FROM _data`
`WHERE fecha_alta >'2009-05-01'`
`AND fecha_baja < '2012-01-01'`
`AND area >500`
`AND cambio_suc='segregated-from'`
`ORDER BY area ASC;`
	`SELECT ?parcel ?area ?refcat`
	`WHERE { ?parcel hco:segregated-from ?x .`
	`?parcel hco:area ?area .`
	`?parcel dc:identifier ?refcat .`
	`FILTER ( ?area > "500"^^xsd:double ) .`
	`FILTER ( ?date < "2012-01-01"^^xsd:dateTime`
	`and ?date > "2009-05-01"^^xsd:dateTime ) }`
	`ORDER BY ASC ( ?area )`
		ninterno, referencia catastral y área de parcelas hija segregadas con un área mayor de 4 ha que existen entre unas fechas determinada, ordenadas de mayor a menor área

`SELECT`
`ST_Within(ST_GeomFromText(`
`'BOX(-7.350 37.125,3.812 42.537)'),`
`ST_Centroid(geometry)), ninterno`
`FROM _data;`
	`SELECT ?parcel ?geo`
	`WHERE { ?parcel rdf:type owl:NamedIndividual.`
	`?parcel geo:geometry ?geo`
	`FILTER(bif:st_contains(bif:st_geomfromtext(`
	`"BOX(-7.350 37.125,3.812 42.537)"),`
	`bif:ST_Geomfromtext(?geo)`
	`)`
	`) }`
		Parcelas dentro de una caja definida

`SELECT`
`ST_Within(ST_GeomFromText(`
`'POLYGON((-7.350 37.125,3.812`
`42.537))'),`
`ST_Centroid(geometry)),ninterno`
`FROM _data`
`WHERE fecha_alta >'2001-05-01'`
`AND fecha_baja < '2012-01-01'`
`AND area >4000`
`AND cambio_suc='segregated-from'`
`ORDER BY area ASC;`
	`SELECT ?parcel`
	`WHERE { ?parcel hco:segregated-from ?x .`
	`?parcel geo:geometry ?geo .`
	`?parcel hco:area ?area .`
	`FILTER ( ?area > "4000"^^xsd:double ) .`
	`FILTER ( ?date < "2012-01-01"^^xsd:dateTime`
	`and ?date > "2001-05-01"^^xsd:dateTime )`
	`FILTER(bif:ST_Within(bif:ST_Geomfromtext(?geo),`
	`bif:ST_Geomfromtext(`
	`"BOX(-7.350 37.125,3.812 42.537)" ),`
	`0.001)`
	`}`
	`ORDER BY ASC ( ?area )`
		Elegir parcelas hijas en función de su área, su fecha de validez y su geometría

Como se puede apreciar la lógica de ambos lenguajes es muy similar, si bien SPARQL es más difícil de entender al principio, después de las primeras consultas se comprende perfectamente, más si ya se está familiarizado con otros lenguajes de consulta.

5. Resultados

Se procede a continuación a enumerar todos los resultados que se han ido obteniendo a lo largo del presente estudio.

En primer lugar y a pesar de no ser algo tangible, son dignos de mención los conocimientos adquiridos sobre ontologías, bases de datos, programas, herramientas y Java durante las fases de estudio y desarrollo del proyecto.

Tras el estudio de las ontologías y la base de datos de linaje catastral se pudo crear un vocabulario, basado en JDO, creado por Lacasta et al. (2014), que lo ampliaba. Mediante esta técnica se han conseguido realizar algunos de los objetivos que busca la web semántica. El presente vocabulario es un paso más hacia una ontología completa sobre las divisiones administrativas y catastrales españolas en caso de que trabajos posteriores se apoyen sobre el mismo.

Una vez comenzada la manipulación de los datos se ha conseguido realizar satisfactoriamente la carga y adaptación de una base de datos "externa" en postgreSQL. Se utilizó una base de datos a la que no se le había realizado ningún estudio previo y de la que no se tenía conocimiento y se llegó a optimizar para el objetivo que se perseguía.

Con la base de datos modificada y exportada en CSV y el vocabulario ontológico implementado sólo restaba unir ambos aspectos del estudio. Para ello se desarrolló un aplicación, en lenguaje Java, capaz de leer la cabecera de la ontología y los datos en CSV (en un orden predeterminado) y guardar como resultado un archivo en RDF que contiene la base de datos semántica de linaje catastral.

Se ha logrado la puesta en marcha un servidor local Virtuoso con el que poder administrar la base de datos.

Finalmente, para comprobar la funcionalidad de las ontologías y la corrección de los datos guardados, se ha realizado una batería de consultas SPARQL a la base de datos semántica mediante la herramienta SPARQL de Virtuoso, que han tenido su homónimo en SQL para la base de datos relacional utilizando la herramienta pgAdmin.

6. Conclusiones y trabajos futuros

Tras el estudio e implementación de una ontología se puede afirmar que existe una gran amplitud de aproximaciones en el campo de los modelos de base de datos y ontologías, que reflejan el esfuerzo dedicado a este área de investigación para desarrollar un sistema de información espaciotemporal con un manejo de datos, análisis y capacidades de consulta mejorados. Este esfuerzo, sin embargo, es también la contrapartida de las ontologías, ya que no suelen existir soluciones estandarizadas, lo cual dificulta su objetivo principal: la interoperabilidad. Evaluar todas las aproximaciones relevantes existentes en el momento de aplicar una ontología, supone un gran esfuerzo y una tarea muy laboriosa.

El estudio de ontologías superiores sobre divisiones administrativas geográficas, a pesar de ser un campo muy extendido, ha demostrado que existen muy pocas definidas, difíciles de encontrar. Este problema puede radicar en la dificultad de crear un marco superior para un campo tan variable, espacial y temporalmente, como las divisiones administrativas. Algunos modelos de divisiones administrativas pueden ser más cercanos que otros, pero normalmente cada modelo tiene sus peculiaridades.

Los buscadores de ontologías sirven como apoyo para definir elementos puntuales, pero no una ontología completa ya que se basan en algoritmos que buscan palabras y ordenan los resultados por popularidad. La búsqueda de una clase concreta que fuera adecuada para el estudio sólo arrojaba ontologías demasiado genéricas que no podía ajustarse con las peculiaridades de la base de datos de linaje catastral; la búsqueda de propiedades concretas daba como resultado ontologías diferentes para cada término cuya definición no era suficiente para el objetivo que se buscaba.

Los listados de ontologías han demostrado ser más útiles ya que agrupaban ontologías completas temáticamente, siendo éstas más completas y concretas que las que ofrecen los buscadores. Su contrapunto es que también dificulta enormemente la búsqueda, ya que al carecer de un buscador, el usuario lo ha de hacer manualmente, aumentando considerablemente la cantidad de trabajo.

La ausencia de una herramienta capaz de aplicar la ontología deseada a una base de datos relacional dada es un freno para el desarrollo de las ontologías, muchos equipos de investigación podrían no contar con un programador para crear un programa distinto para cada implementación y abandonar el campo ontológico.

Se ha comprobado que la base de datos semántica funciona al mismo nivel en cuanto a procesado de datos y estabilidad que la base de datos relacional: las consultas SPARQL y sus correspondientes consultas equivalentes SQL dan como resultado los mismos datos con un tiempo de respuesta muy similar.

El lenguaje SPARQL, a pesar de parecerse a SQL, tiene una lógica más compleja y necesita de mayor estudio por parte del usuario, pero las ventajas que aporta la semántica frente a la base relacional superan ampliamente este inconveniente.

Dentro de estas ventajas se han encontrado:

- Modificar una propiedad o añadir una nueva no requeriría de guardar un nuevo registro
    - hace más funcional la base de datos
    - más procesal
    - reduce el tamaño de la base de datos
- Permite realizar consultas sobre la genealogía y trazabilidad de las parcelas

Por otro lado se cuenta también las ventajas otorgadas por la web semántica:

- Mejor organización de los datos
- Interoperabilidad
- Datos “entendibles” por los ordenadores

Como desventajas encontradas en para la implementación de bases de datos semánticas se pueden citar:

- El costoso y laborioso proceso que conlleva la transformación a bases de datos semánticas
- Problemas técnicos
- Falta de desarrollo de aplicaciones para implementar ontologías
- Complejidad de lenguaje y codificación

A pesar de las desventajas, se cree que el trabajo invertido en la implementación de los vocabularios, compensa enormemente al usuario al crear una base de datos que finalmente será mucho más asequible y entendible para todo el mundo.

Lamentablemente la versión de la aplicación Virtuoso utilizada tenía implementadas pocas funciones espaciales. Se cree que dichas funciones son de gran utilidad y que seguramente futuras versiones sean mucho más completas en este sentido. A pesar de la gran utilidad que ha demostrado en las consultas básicas, este estudio debe rechazar la versión 7.10.3207 de Virtuoso por no cumplir con uno de los requerimientos básicos para el mismo: la realización de consultas espaciales complejas.

En cuanto a este trabajo, la línea de desarrollo que, se cree, debería seguir es la siguiente:

Primero, hacer un análisis de las capacidades de los SPARQL *end-points* para consultas espaciotemporales de distintos tipos.

Posteriormente la realización de una aplicación web o mashup similar a la encontrada en el trabajo de fin de máster de Sinde (2014) con el fin de facilitar las consultas del usuario mostrar los resultados de manera mucho más útil y entendible.

La figura 16 presenta una maqueta del aspecto que podría tener un visor de datos de linaje catastral. Este moqup (o maqueta) también se puede encontrar en línea en el siguiente enlace: https://moqups.com/AlvBch/N0DeHKsh

Otra línea de investigación podría ser el desarrollo del vocabulario abarcando más aspectos de las divisiones administrativas y catastrales del territorio español.

Finalmente, otra línea de investigación relacionada con las ontologías y los vocabularios es la implementación de un programa que sea capaz de recoger la información de una base de datos y, mediante la indicación del significado de cada campo, sea capaz de implementar automáticamente cualquier ontología, guardandola en los formatos habituales (OWL, RDF…).

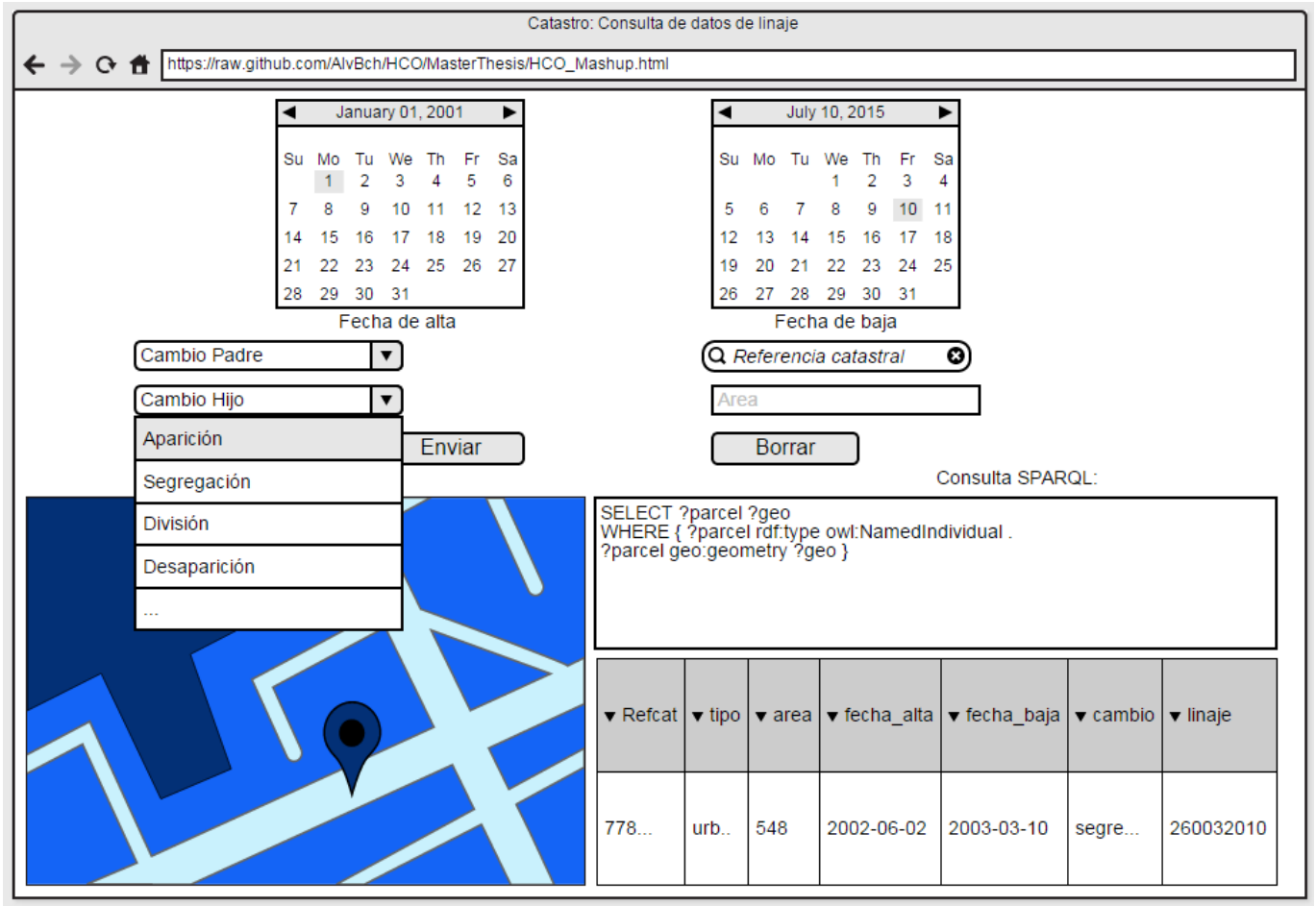

Figura 16. Mashup para consulta de datos de linaje catastral

Bibliografía

1. Apache Jena (2011): "Apache Jena – SPARQL Tutorial" http://jena.apache.org/tutorials/sparql.html Accedido el 2 de julio de 2015.
2. Agarwal, P. (2005): "Ontological considerations in GIScience", *International Journal of Geographical Information Science*, Vol.19, No.5, pp. 501-536.
3. COLORE (2009): "COLORE Overview" http://stl.mie.utoronto.ca/colore/ Accedido el 27 de junio de 2015.
4. DAML (2006): "DAML.org" http://www.daml.org Accedido el 27 de junio de 2015.
5. Fu, G.; Jones, C.B. y Abdelmonty, A.I. (2005): "Building a Geographical Ontology for Intelligent Spatial Search on the Web", *Proceedings of LASTED International Conference on Databases and Applications*, pp. 167-172.
6. Gantner, F. (2011): "A Spatiotemporal Ontology for the Administrative Units of Switzerland", *Tesis doctoral*. University of Zurich en colaboración con la Swiss Federal Institute for Forest, Snow and Landscape Research.
7. Gruber, T. R. (1993): "A translation approach to portable ontologies", *Knowledge Acquisition*, 5(2), pp. 199-220.
8. Gruber, T. R. (2009): "Ontology", *Encyclopedia of Databbase Systems*, Springer-Verlag.
9. Guarino, N. (1998): "Formal Ontology and Information Systems", *Formal Ontology in Information Systems, Proceedings of FOIS'98, Trento, Italy, 6-8*. Amsterdam, IOS Press, pp. 3-15.
10. Incubator Group, W3C (2009): "A Survey of Current Approaches for Mapping of Relational Databases to RDF" http://www.w3.org/2005/Incubator/rdb2rdf/RDB2RDF_SurveyReport.pdf Accedido el 2de julio de 2015.
11. Lacasta, J.; Lopez-Pellicer, F.J.; Florczyk, A.; Zarazaga-Soria, F.J. y Nogueras-Iso, J. (2014): "Population of a spatio-temporal knowledge base for jurisdictional domains", *International Journal of Geographical Information Science*, Vol.28, No.9, pp.1964-1987.
12. Linked Open Vocabularies, LOV (2013): "Linked Open Vocabularies/About"

http://lov.okfn.org/dataset/lov Accedido el 26 de junio de 2015.

13. Martínez-Páez, F. J. (2006): "Trabajar con ficheros CSV | Adictos al trabajo" http://www.adictosaltrabajo.com/tutoriales/csv/ Accedido el 2 de julio de 2015.

14. Moreno, P.; Iturrioz, T. y Martinez, S. (2014): "Metodología para deducir relaciones de linaje en el Catastro de España", *GeoFocus (Artículos)*, No.14, pp.275-300.

15. NeOn Project (2010): "Ontologies - NeOn Project" http://www.neon-project.org/nw/Ontologies Accedido el 27 de junio de 2015.

16. OSGeo (2010): "GeoKettle – OSGeo-Live 8.5 Documentation" http://live.osgeo.org/es/overview/geokettle_overview.html Accedido el 28 de junio de 2015.

17. PgAdmin (2014): "pgAdmin: Documentation" http://www.pgadmin.org/docs/ Accedido el 2 de julio de 2015.

18. Portal de la Dirección General de Catastro (2007): "¿Qué es Catastro?" http://www.catastro.meh.es/default.asp Accedido el 23 de junio de 2015.

19. PostgreSQL (2015): "PostgreSQL: Documentation: 7.1: PostgreSQL 7.1.3 Documentation" http://www.postgresql.org/docs/7.1/static/postgres.html Accedido el 2 de julio de 2015.

20. PURL (2015): "PURL Home Page" https://purl.org/docs/index.html Accedido el 27 de junio de 2015.

21. Real Academia Española, R.A.E. (2014): "Diccionario de la Real Academia Española" http://www.rae.es/recursos/diccionarios/drae Accedido el 11 de junio de 2015.

22. Semantic Web (2012): "Ontology" http://semanticweb.org/wiki/Ontology Accedido el 24 de junio de 2015.

23. Sinde, I.; Alcarria, R. P.; Manso, M. A. (2014): "Estudio de las posibilidades de los datos abiertos enlazados (Linked Open Data) para la realización de mashups de ámbito geográfico", *Trabajo de fin de* máster, Universidad Politécnica de Madrid.

24. Shirky, C. (2005): "Ontology is Overrated: Categories, Links and Tags" http://www.shirky.com/writings/ontology_overrated.html Accedido el 24 de junio de 2015.

25. Spatially-Aware Information Retrieval on the Internet, SPIRIT (2007): "SPIRIT – Spatially-Aware Information Retrieval on the Internet" http://www.geo-spirit.org/index.html Accedido el 26 de junio de 2015.

26. Swoogle (2005): "Swoogle Semantic Web Search Engine" http://swoogle.umbc.edu Accedido el 26 de junio de 2015

27. Villalobos, J. (2011): "Manejo de archivos en java: escribir en un archivo de texto (.txt) | Codigoprogramacion" http://codigoprogramacion.com/cursos/java/130-manejo-de-archivos-en-java-escribir-en-un-archivo-de-texto-txt.html#.VPjV0F3WsXt Accedido el 2 de julio de 2015.

28. Villalobos, J. (2011): "Manipulación de archivos: lectura de un archivo de texto en java | Codigoprogramacion" http://codigoprogramacion.com/cursos/java/110-manipulacion-de-archivoslectura-de-un-archivo-de-texto-en-java.html Accedido el 2 de julio de 2015.

29. Virtuoso (2015): "OpenLink Virtuoso Homepage" http://virtuoso.openlinksw.com Accedido el 2 de julio de 2015.

30. Watson (2008): "Watson Semantic Web Search" http://watson.kmi.open.ac.uk/WatsonWUI/ Accedido el 26 de junio de 2015.

31. W3C (2004): "OWL Web Ontology Language Reference" http://www.w3.org/TR/owl-ref/ Accedido el 27 de junio de 2015.

32. W3C (2004): "Guía Breve de la Web Semántica" http://www.w3c.es/Divulgacion/GuiasBreves/WebSemantica Accedido el 24 de junio de 2015.

33. W3C (2008): "Ontology" http://www.w3.org/standards/semanticweb/ontology Accedido el 25 de junio de 2015.

34. W3C (2008): "SPARQL Query Language for RDF" http://www.w3.org/TR/rdf-sparql-query/ Accedido el 2 de julio de 2015.

35. W3C (2009): "Guía Breve de Linked Data" http://www.w3c.es/Divulgacion/GuiasBreves/LinkedData Accedido el 24 de junio de 2015.

36. W3C (2009): "OWL 2 Web Ontology Document Overview" http://www.w3.org/TR/2009/WD-owl2-overview-20090327/ Accedido el 27 de junio de 2015.

37. W3C (2009): "OWL Web Ontology Language Reference" http://www.w3.org/TR/owl-ref/ Accedido el 21 de junio de 2015.

38. W3C (2012): "W3C XML Schema Definition Language (XSD) 1.1 Part 1: Strutures" http://www.w3.org/TR/xmlschema11-1/ Accedido el 27 de junio de 2015.

39. W3C (2013): "List of Ontologies" http://www.w3.org/wiki/Lists_of_ontologies Accedido el 27 de junio de 2015.

40. W3C (2014): "RDF – Semantic Web Standards" http://www.w3.org/RDF/ Accedido el 27 de junio de 2015.

41. W3C (2014): "RDF Schema 1.1" http://www.w3.org/TR/rdf-schema/ Accedido el 27 de junio de 2015.

42. W3C (2015): "ISA Programme Location Core Vocabulary" http://www.w3.org/ns/locn Accedido el 27 de junio de 2015.

43. Yuan, M. (2008): "Adding Time into Geographic Information System Databases", *The Handbook of Geographic Information Science*, publicado online por Blackwell Publishing Ltd.

ANEXO I

El presente anexo muestra ejemplos de las tablas que se han manejado durante el estudio.

Las tablas se muestran divididas en varias partes ya que de presentarse en una sola parte sería ilegíble.

Tabla 7. Datos de entrada "parcelas"

pcat1 (text)	pcat2 (text)	numerodup (text)	hoja (text)	tipo (text)	refcat (text)	fecha_inic (text)	fecha_fin (text)	the_geom (geometry(Geometry,4258)	coory (double precision)
28001A0	0500076		A	R	28001A00500076	09/22/2009		0106000020A210000001[…]	4547166.11

via (integer)	numero (integer)	numsymbol (integer)	area (double precision)	fechaalta (text)	fechabaja (text)	ninterno (bigint)	mapa (integer)	delegacio (integer)	municipio (integer)	masa (text)
0	0	0	2322	20090922	99999999	260039137	140	28	1	005

parcela (text)	coorx (double precision)	ID [PK](integer)
00076	448256.05	1

Tabla 8. Datos de entrada "linaje"

gid [PK](serial)	__gid (numeric(10,0))	padre (character varying(14))	gid_hijo (numeric(10,0))	padre (character varying(14))	area_padre (numeric(10,0))	f_ini_padr (character varying(10))
6	1	1125	1125	6280001VL4368S	54872	01/11/2005

f_fin_padr (character varying(10))	gid_hijo (numeric(10,0))	hijo (character varying(14))	area_hijo (numeric(10,0))	f_ini_hijo (character varying(10))	f_fin_hijo (character varying(10))
10/28/2011	429	6280001VL4368S	38934	10/28/2011	10/07/2013

cambio (numeric(10,0))	tipo (character varying(2))
104	S

Tabla 9. Datos de salida CSV para transformar a RDF

ninterno [PK](bigint)	refcat (text)	mapa (integer)	delegacion (integer)	municipio (integer)	tipo (text)	area (double precision)	the_geom (text)	fecha_alta (date)
260031948	7787232VL4478N	334	28	1	urban	548	MULTIPOLYGON(((-3.61721371459268 […]	2002-06-27

fecha_baja (date)	cambio_pre (text)	cambio_suc (text)	sucesor (text)	predecesor (text)
2003-03-18	segregates		260032010 260032011	

En esta tabla es crucial el orden de los datos ya que será el que tome la aplicación Java para crear la base de datos semántica